Shielding from Space Radiations

Annual Technical Report

Principal Investigator: Dr. C. Ken Chang
Forooz F. Badavi
and
Dr. Ram K. Tripathi

Period: December 1, 1992 through June 1, 1993

CHRISTOPHER NEWPORT UNIVERSITY
NEWPORT NEWS, VIRGINIA 23606-2998

NASA GRANT NUMBER 1-1286

ABSTRACT

This Progress Report covering the period of December 1,1992 to June 1,1993 presents the development of an analytical solution to the heavy ion transport equation in terms of Green's function formalism. The mathematical development results are recasted into a highly efficient computer code for space applications. The efficency of this algorithm is accomplished by a nonperturbative technique of extending the Green's function over the solution domain. The code may also be applied to accelerator boundary conditions to allow code validation in laboratory experiments. Results from the isotopic version of the code with 59 isotopes present for a single layer target material, for the case of an Iron beam projectile at 600 MeV/nucleon in water is presented. A listing of the single layer isotopic version of the code is included.

INTRODUCTION

Future NASA missions will be limited by exposure to space radiations unless adequate shielding is provided to protect men and equipments from such radiations. Adequate methods required to estimate the damage caused by such radiations behind various shields can be evaluated prior to commitment to such missions.

From the inception of the Langley Research Center heavy ion (HZE) shielding program (refs. 1-3), there has been a continued, close relationship between code development and laboratory experiment (ref. 3). Indeed, the current research goal is to provide computationally efficient high charge and energy ion (HZE) transport codes which can be validated with laboratory experiments and subsquently applied to space engineering design. In practice, two streams of code development have prevailed due to the strong energy dependence of necessary atomic/molecular cross sections and the near singular nature of the laboratory beam boundary conditions (refs. 4-6). The atomic/molecular cross section dependence is adequately dealt with by using the methods of Wilson and Lamkin (ref. 7), allowing efficient numerical procedures to be developed for space radiations (refs. 6,8-10). Although these codes could conceivably be applied to the laboratory validation, methods to control truncation and discretization errors would bear little resemblance to the space radiation codes attempting to be validated. Clearly, a radical reorientation is required to achieve the validation goals of the current NASA space radiation shielding program, and such an approach is the main thrust of this research and is briefly described below.

A useful technique in space radiation shielding is the use of the impulse response Green's function (refs. 11,12), which satisfies the Boltzman equation of the form

$$\left[\vec{\Omega}\cdot\vec{\nabla} - \frac{\partial}{\partial E}\tilde{S}_j(E) + \sigma_j\right] G_{jm}(E, E_o, \vec{\Omega}, \vec{x})$$

$$= \sum_k \int \sigma_{jk}(E, E', \vec{\Omega}, \vec{\Omega}')\, G_{km}(E', E_o, \vec{\Omega}', \vec{x}) d\vec{\Omega}' dE' \quad (1)$$

where G_{jm} reduces to a monoenergetic unidirectional function at the boundary, $\tilde{S}_j(E)$ is the stopping power, σ_j is the total cross section, and σ_{jk} is the inclusive differential cross section. An arbitrary solution to the Boltzman equation within a closed convex region can be written as

$$\phi_j(E, \vec{\Omega}, \vec{x}) = \sum_m \int G_{jm}(E, E', \vec{\Omega} - \vec{\Omega}', \vec{x} - \vec{\Gamma})$$

$$\times\ f_m(E', \vec{\Omega}', \vec{\Gamma}) d\vec{\Omega}' dE' d\vec{\Gamma} \quad (2)$$

where $f_m(E', \vec{\Omega}', \vec{\Gamma})$ is the incident flux at the boundary (ref. 11). Since transport problem is formulated in terms of a single Green's function algorithm, the validation of the Green's function in the laboratory meets the objective of having a space validated code. Since there is hope of a Green's function based on an analytical solution of the Boltzmann equation (ref. 13), the resulting evaluation of the shield properties should be computationally efficient.

The first step in this process is to develop an equivalent Green's function algorithm in one dimension to match the current capability in space radiation transport calculation (refs. 6,14). The algorithm is based on the closed form solution to the one dimensional equation

$$\left(\frac{\partial}{\partial x} - \frac{\partial}{\partial E}\tilde{S}_j(E) + \sigma_j\right) G_{jm}(E, E_o, x)$$

$$= \sum_k \int \sigma_{jk}(E, E') G_{km}(E', E_o, x) dE' \qquad (3)$$

for a monoenergetic beam at the boundary. The probality of validation for ^{20}Ne beams of this algorithm (with multiple scattering corrections) has already shown good correlation (refs. 5,15), but improvements in the nuclear data base are required for achieving higher correlations with experiment. If considerations are restricted to multiple charged ions then the right hand side of equation (3) can be further reduced to

$$\left[\frac{\partial}{\partial x} - \frac{\partial}{\partial E}\tilde{S}_j(E) + \sigma_j\right] G_{jm}(E, E_o, x)$$

$$= \sum_k \sigma_{jk} G_{km}(E, E_o, x) \qquad (4)$$

for which a solution is presented below.

APPROXIMATE GREENS'S FUNCTION

Equation (4) can be simplified by transforming the energy into the the residual range as

$$r_j = \int_0^E dE'/\tilde{S}_j(E') \qquad (5)$$

and defining new field variables as

$$\psi_j(x, r_j) = \tilde{S}_j(E)\phi_j(x, E) \qquad (6)$$

$$\mathcal{G}_{jm}(x, r_j, r'_m) = \tilde{S}_j(E) G_{jm}(x, E, E') \qquad (7)$$

so that equation (4) becomes

$$\left[\frac{\partial}{\partial x} - \frac{\partial}{\partial r_j} + \sigma_j\right] \mathcal{G}_{jm}(x, r_j, r'_m)$$

$$= \sum_k \frac{\nu_j}{\nu_k} \sigma_{jk} \mathcal{G}_{km}(x, r_j, r'_m) \qquad (8)$$

with boundary condition

$$\mathcal{G}_{jm}(0, r_j, r'_m) = \delta_{jm}\, \delta(r_j - r'_m) \qquad (9)$$

and

$$\psi_j(x, r_j) = \sum_m \int_0^\infty \mathcal{G}_{jm}(x, r_j, r'_m) f_m(r'_m) dr'_m \qquad (10)$$

The solution to equation (8) may be written as

$$\mathcal{G}_{jm}(x, r_j, r'_m) = \sum_i \mathcal{G}^{(i)}_{jm}(x, r_j, r'_m) \qquad (11)$$

where zeroth order term of equation (11) is

$$\mathcal{G}^{(o)}_{jm}(x, r_j, r'_m) = g(j)\, \delta_{jm}\, \delta(x + r_j - r'_m) \qquad (12)$$

and the first order term of equation (11) is

$$\mathcal{G}^{(1)}_{jm}(x, r_j, r'_m) \approx \frac{\nu_j\, \sigma_{jm}\, g_{(j,m)}}{x(\nu_m - \nu_j)} \qquad (13)$$

with the condition that $\mathcal{G}^{(1)}_{jm}(x, r_j, r'_m)$ is zero unless

$$\frac{\nu_j}{\nu_m}(r_j + x) \le r'_m \le \frac{\nu_i}{\nu_m} r_j + x \qquad (14)$$

The second order terms of equation (11) are

$$\mathcal{G}^{(2)}_{jm}(x, r_j, r'_m) \approx \sum_k \frac{\sigma_{jk}\, \sigma_{km}\, g(j, k, m)}{r'_{mu} - r'_{m\ell}} \qquad (15)$$

with the condition that $\mathcal{G}^{(2)}_{jm}(x, r_j, r'_m)$ are nonzero for

$$r'_{m\ell} \le r'_m \le r'_{mu} \qquad (16)$$

where

$$r'_{mu} = \left\{ \begin{array}{ll} \frac{\nu_j}{\nu_m} r_j + x & (\nu_m > \nu_k > \nu_j) \\ \frac{\nu_j r_j + \nu_k x}{\nu_m} & (\nu_k > \nu_m > \nu_j) \\ \frac{\nu_j}{\nu_m} r_j + x & (\nu_m > \nu_j > \nu_k) \end{array} \right\} \qquad (17)$$

and

$$r'_{m\ell} = \left\{ \begin{array}{ll} \frac{\nu_j}{\nu_m}(r_j + x) & (\nu_m > \nu_k > \nu_j) \\ \frac{\nu_j}{\nu_m}(r_j + x) & (\nu_k > \nu_m > \nu_j) \\ \frac{\nu_j r_j + \nu_k\, x}{\nu_m} & (\nu_m > \nu_j > \nu_k) \end{array} \right\} \qquad (18)$$

The third order terms of equation (11) are

$$g^{(3)}_{jm}(x, r_j, r'_m) \approx \sum_{k,\ell} \frac{\sigma_{jk}\, \sigma_{k\ell}\, \sigma_{\ell m}\, g(j,k,\ell,m)}{r'_{mu} - r'_{m\ell}} \qquad (19)$$

and similarly for higher order terms. In the above the g's of n arguments are given by

$$g(j) = e^{-\sigma_j\, x} \qquad (20)$$

and

$$g(j_1, j_2, \cdots j_n, j_{n+1})$$

$$= \frac{g(j_1, j_2 \cdots j_{n-1}, j_n) - g(j_1, j_2 \cdots, j_{n-1}, j_{n+1})}{\sigma_{j_{n+1}} - \sigma_{j_n}} \qquad (21)$$

In terms of above, the solution to equation (4) may be written as

$$\psi_j(x, r_j) = e^{-\sigma_j x} f_j[R_j^{-1}(r_j + x)]$$

$$+ \sum_{m,\ell} g_{jm}^{(i)}(x) \Big\{ F_m[R_m^{-1}(r'_{m\ell})]$$

$$- F_m[R_m^{-1}(r'_{mu})] \Big\} \qquad (22)$$

where

$$g_{jm}^{(i)}(x) =$$

$$\sum_{j_1 j_2 \cdots j_{n-2}} \frac{\sigma_{jj}, \sigma_{j_1 j_2} \cdots \sigma_{j_{n-2} m}\, g(j, j_1 j_2 \cdots j_{n-2}, m)}{\Delta^{(i)}} \qquad (23)$$

for i=1, the denominator of equation (23) is

$$\Delta^{(1)} = x\,(\frac{\nu_m}{\nu_j} - 1) \qquad (24)$$

and for i>1, the denominator becomes

$$\Delta^{(i)} = \left\{ \begin{array}{ll} x\left(1 - \frac{\nu_j}{\nu_m}\right) & (\nu_m > \nu_k > \nu_j) \\ x\left(\frac{\nu_k}{\nu_m} - \frac{\nu_j}{\nu_m}\right) & (\nu_k > \nu_m > \nu_j) \\ x\left(1 - \frac{\nu_k}{\nu_m}\right) & (\nu_m > \nu_j > \nu_k) \end{array} \right\} \qquad (25)$$

In equation (22), $F_m(E)$ is the integral flux at the boundary, and is defined as

$$F_m(E) = \int_E^\infty f_m(E')dE' \qquad (26)$$

Implementation of equation (22) can now be accomplished independent of the character of the boundary values $f_m(E')$ and will give accurate results for both space and laboratory applications.

DISCUSSION OF RESULTS

Values of collision related fluxes for 3 depths of water target for a mono-energetic beam of Iron projectile with 600 MeV/nucleon are shown in figures 1 through 3 for both the perturbative and nonperturbative Green's function methods for the direct comparison of the two methods. The first three collision terms and the sum of all collision terms of both theories at a depth of 5 cm of water are shown in figure 1.A through 1.H. The differences in the spectral shape is due to the simplification of the attenuation term in the nonperturbative theory. The nonperturbation terms represnt the average spectrum while perturbation theory retains the spectral shape. Figures 2.A through 2.H and 3.A through 3.H are the corresponding comparison of the two methods at depths of 10 and 15 cm of water. Direct comparsions of figures 1 through 3 shows that the sequence of perturbation terms appear to be converging to a result similar to that of nonperturbative result.

The main advantage of nonperturbative methods are in their computational efficencies. The computational time required for the nonperturbative code is about 10 minutes on VAX 4000 compared to 15, 45, 90 minutes for the 1-st, 2-nd and 3-rd collision terms of the perturbation solution.

Figures 4.A through 4.C show the corresponding differential LET spectrum using the method of reference 16. The highest LET peak is due to the primary beam and the ion fragments. The successive peaks below iron are due to lower atomic weight fragments. Such LET spectra can be compared to experimental measurments directly.

REFERENCES

1. J. W. Wilson and C. M. Costner, "Nucleon and Heavy Ion Total and Absorption Cross Section for Selected Nuclei," NASA TN D-8107, December 1975.

2. J. W. Wilson, "Analysis of the Theory of High Energy Ion Transport," NASA TN D-8381, March 1977.

3. J. W. Wilson, "Depth Dose Relations for Heavy Ion Beams," Va. J. of Sci. 28, 136-138 (1977).

4. J. W. Wilson, "Heavy Ion Transport in the Straightahead Approximation," NASA TP-2178, June 1983.

5. J. W. Wilson, L. W. Townsend, H. B. Bidasaria, W. Schimmerling, M. Wong and J. Howard, " Ne Depth Dose Relations in Water, "Health Physics 46, 1101 (1984).

6. J. W. Wilson and F. F. Badavi, "Methods of Galactic Heavy Ion Transport," Radiat. Res. 108, 231 (1986).

7. J. W. Wilson and S. L. Lamkin, "Perturbation Approximation for Charged Particle Transport in One Dimension," Nucl. Sci. and Eng. 57, 292-299 (1975).

8. J. W. Wilson, L. W. Townsend, S. Y. Chun, S. L. Lamkin, B. D. Ganapol, B. S. Hong, W. W. Buck, F. Khan, F. A. Cucinotta and J. E. Nealy, "BRYNTRN: A Baryon Transport Model, " NASA TP-2887, 1989.

9. J. L. Shinn, J. W. Wilson, M. Weyland and F. A. Cucinotta, "Improvements in the computational Accuracy of BRYNTRN (A Baryon Transport Code," NASA TP-3093, 1991.

10. J. L. Shinn and J. W. Wilson, "An Efficient HZETRN (A Galactic Cosmic Ray Transport Code)," NASA TP-3147, 1991.

11. J. W. Wilson and G. S. Khandelwal, "Dose Approximations in Arbitrary Convex Geometry," Nucl. Tech. 23,298 (1974).

12. J. W. Wilson, S. L. Lamkin, B. D. Ganopal, H. Farhat and L. W. Townsend , "A Hierarchy of HZE Transport Approximations," NASA TM-4118, 1989.

13. J. W. Wilson, L. W. Townsend and B. D. Ganopal, "A Closed Form Solution to the HZE Propagation," Radiat. Res. 122, 223 (1990).

14. J. W. Wilson, S. Y. Chun, F. F. Badavi and L. W. townsend, "HZETRN: A Heavy Ion/Nucleon transport Code for Space Radiations," NASA TP-3146, 1991.

15. W. Schimmerling, "Ground Based Measurments of Galactic Cosmic Ray Fragmentation in Shielding," COSPAR, 1990.

16. J. W. Wilson, F. F. Badavi, "A Study of Generation of Linear Energy Transfer Spectra for Space Radiations," NASA TM-4410, 1992.

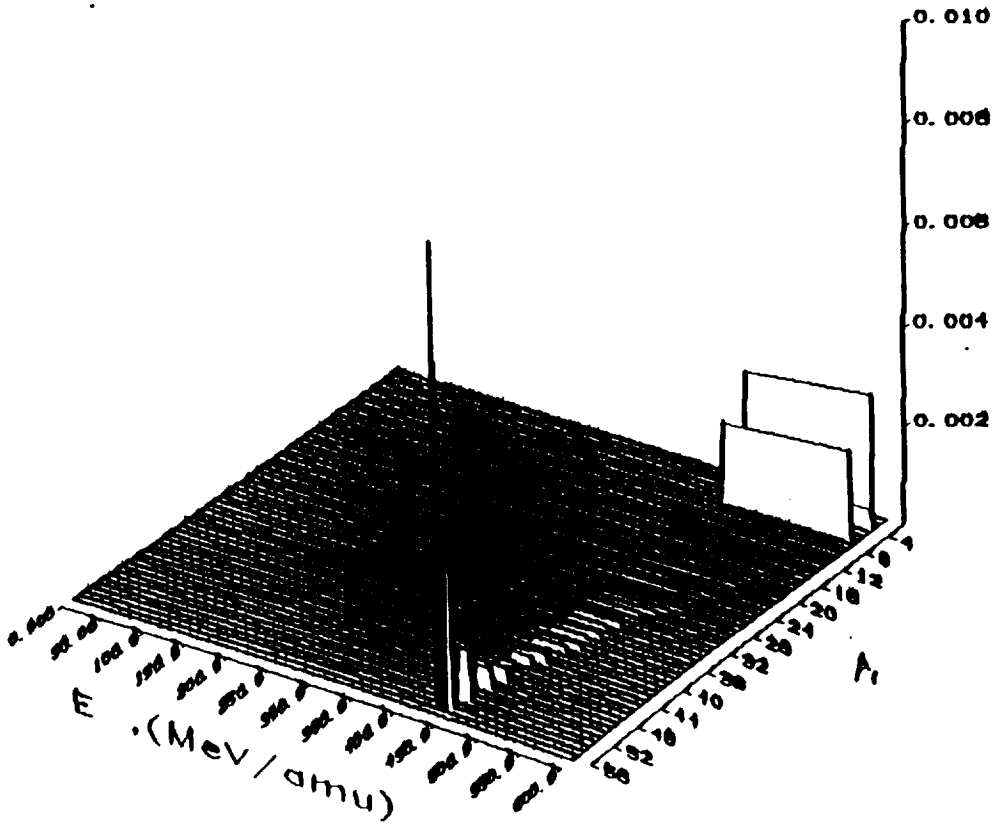

Figure 1.A. 1-st term nonpertubation solution at a depth of 5 cm of water for a 600 MeV/nucleon Iron projectile.

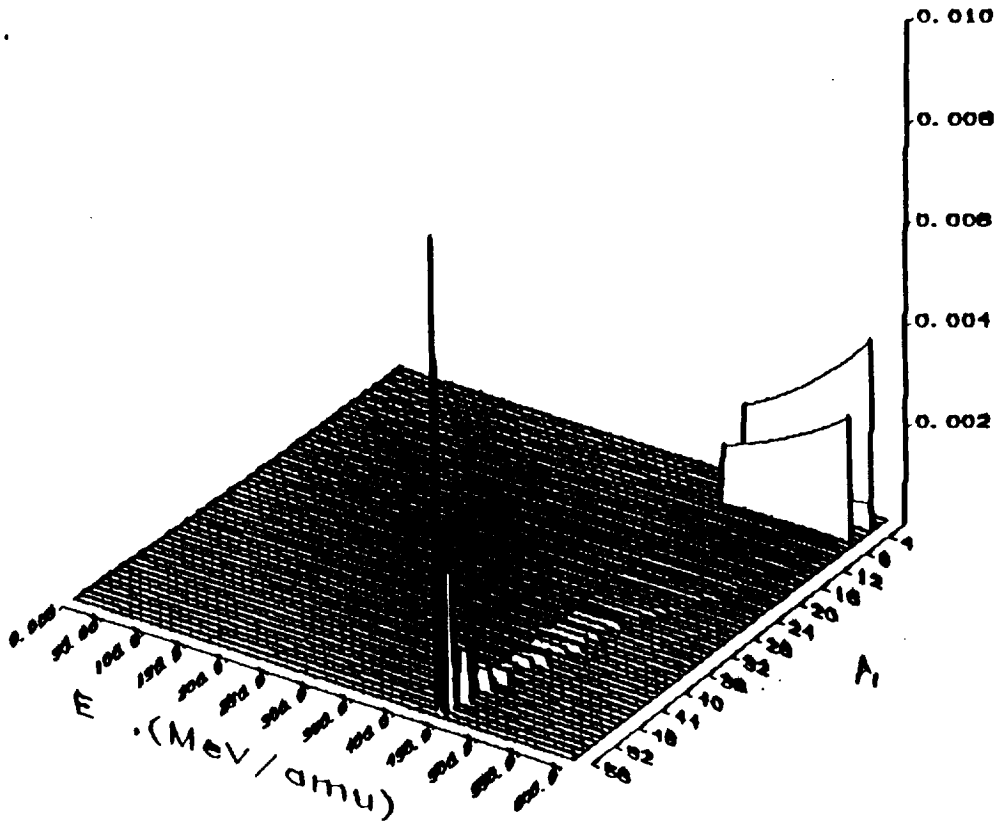

Figure 1.B. 1-st term perturbation solution at a depth of 5 cm of water for a 600 MeV/nucleon Iron projectile.

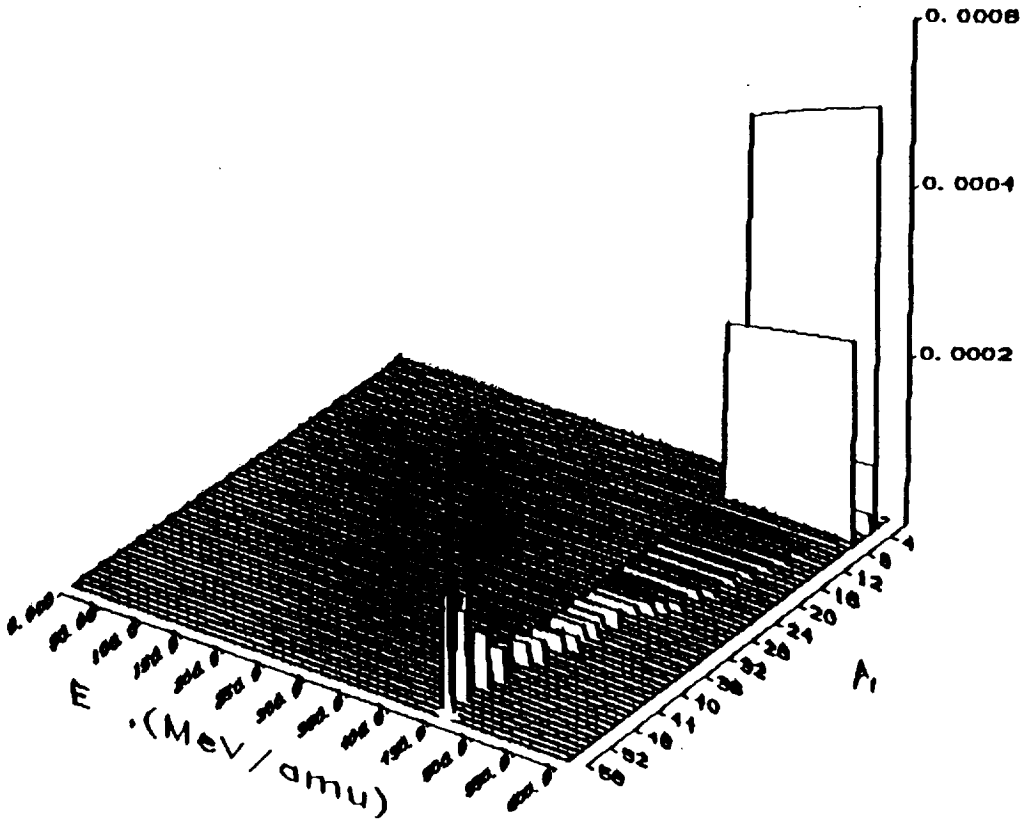

Figure 1.C. 2-nd term nonperturbation solution at a depth of 5 cm of water for a 600 MeV/nucleon Iron projectile.

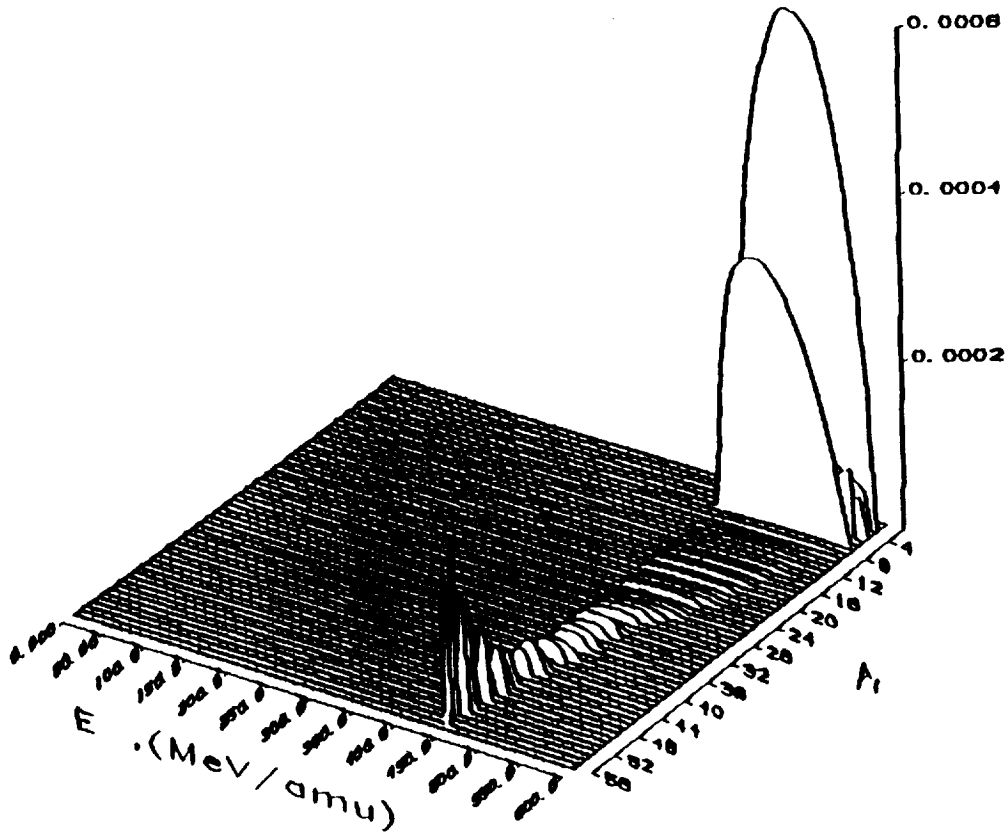

Figure 1.D. 2-nd term perturbation solution at a depth of 5 cm of water for a 600 MeV/nucleon Iron projectile.

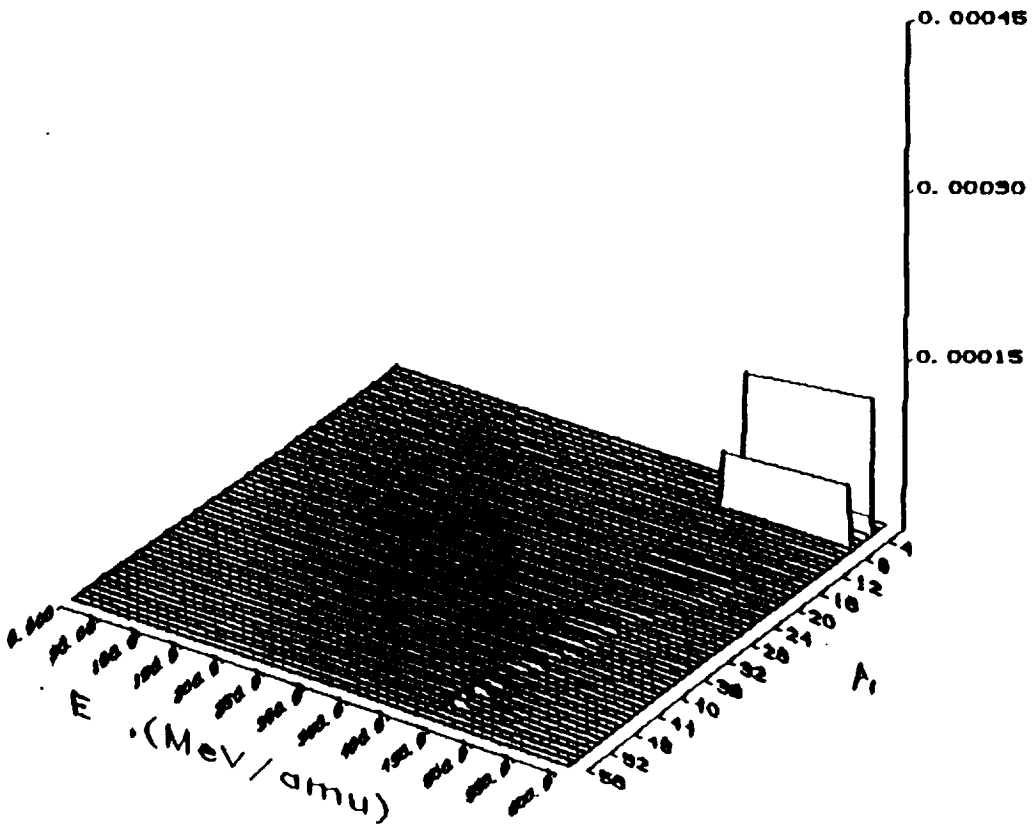

Figure 1.E. 3-rd term nonperturbation solution at a depth of 5 cm of water for a 600 MeV/nucleon Iron projectile.

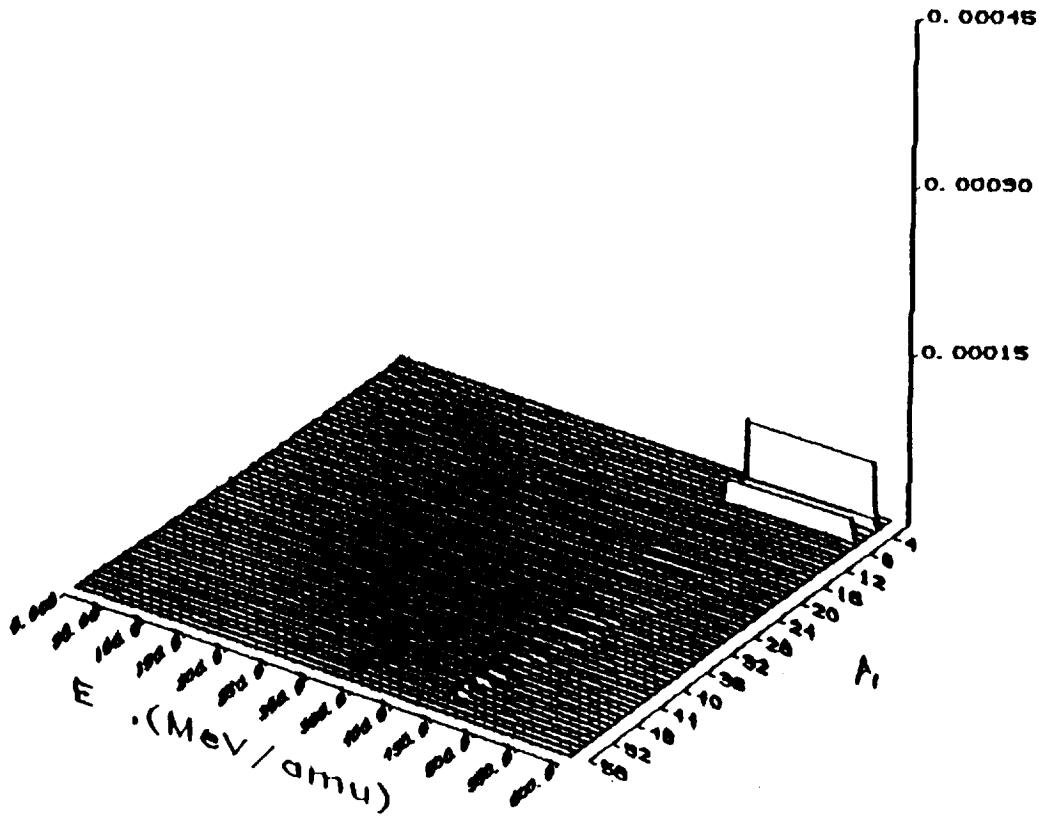

Figure 1.F. 3-rd term perturbation solution at a depth of 5 cm of water for a 600 MeV/nucleon Iron projectile.

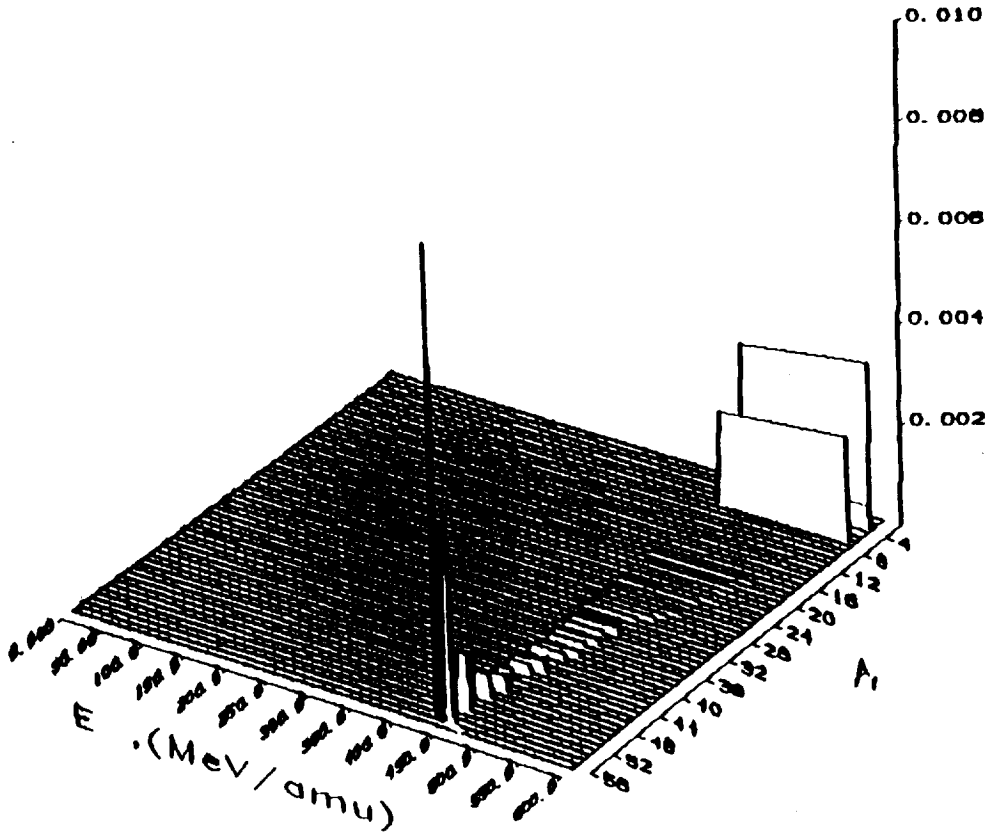

Figure 1.G. All terms nonperturbation solution at a depth of 5 cm of water for a 600 MeV/nucleon Iron projectile.

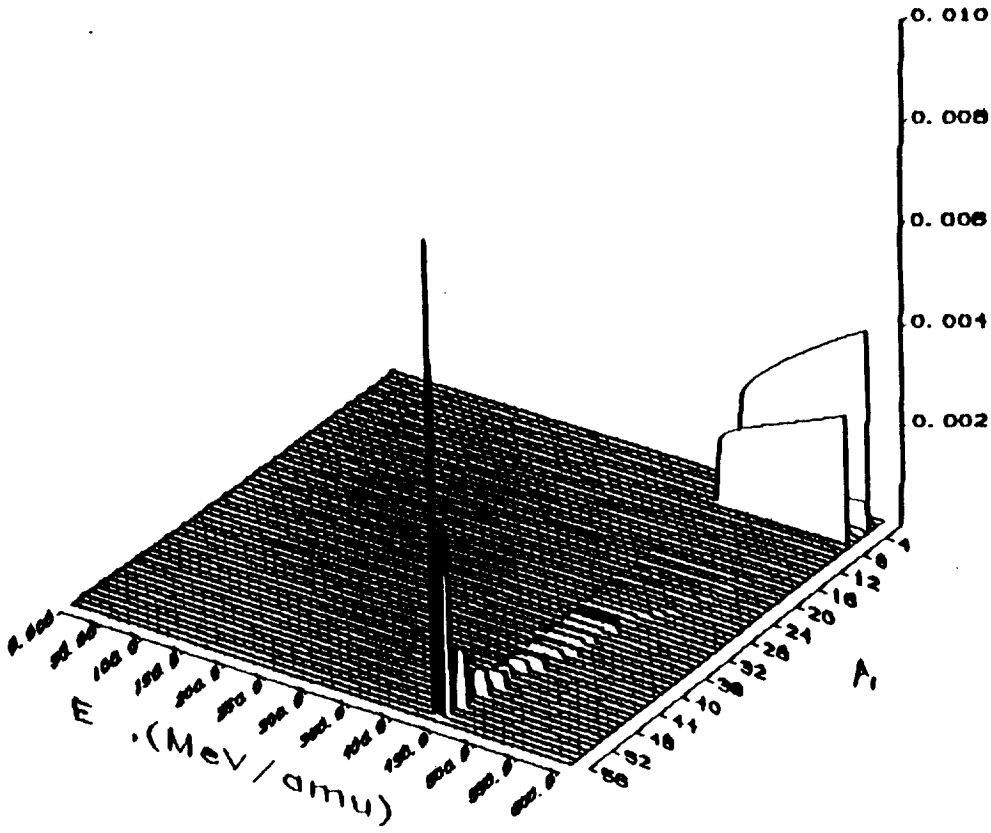

Figure 1.H. All terms perturbation solution at a depth of 5 cm of water for a 600 MeV/nucleon Iron projectile.

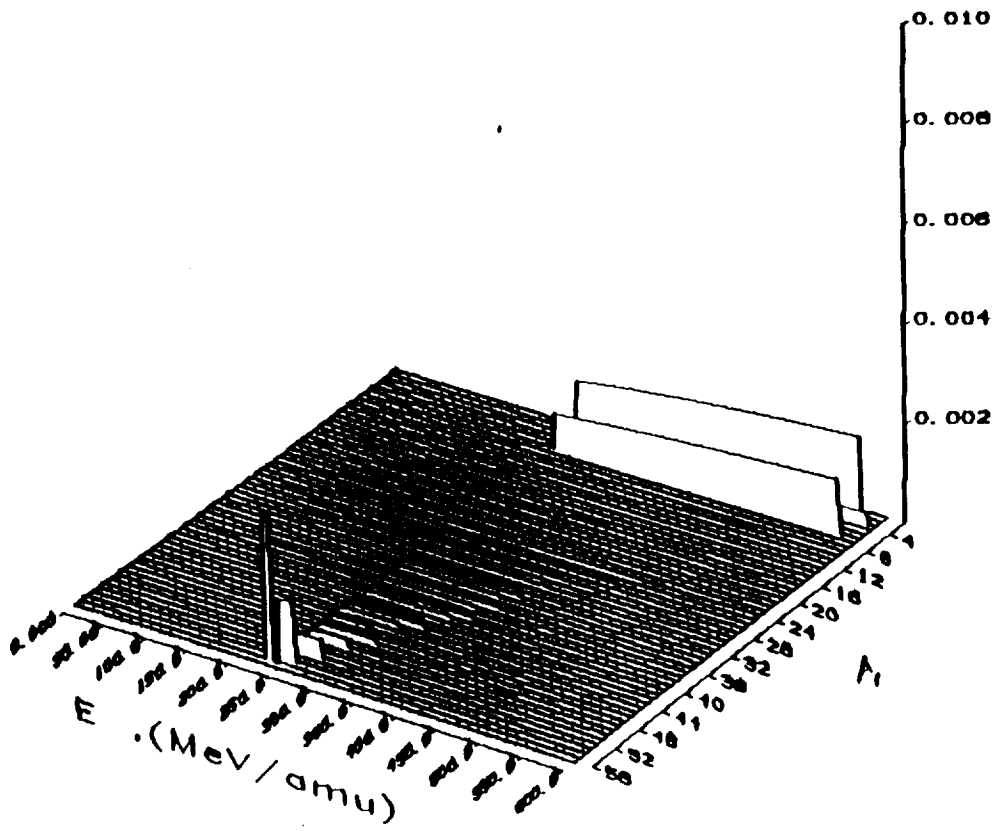

Figure 2.A. 1-st term nonperturbation solution at a depth of 10 cm of water for a 600 MeV/nucleon Iron projectile.

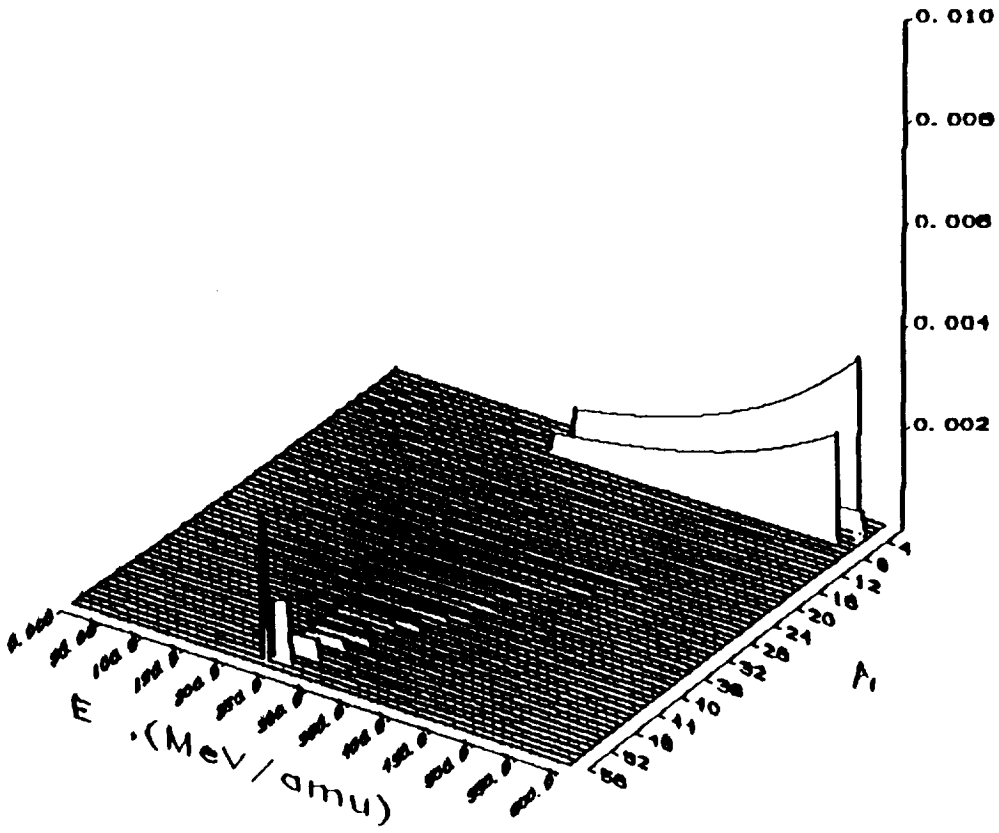

Figure 2.B. 1-st term perturbation solution at a depth of 10 cm of water for a 600 MeV/nucleon Iron projectile.

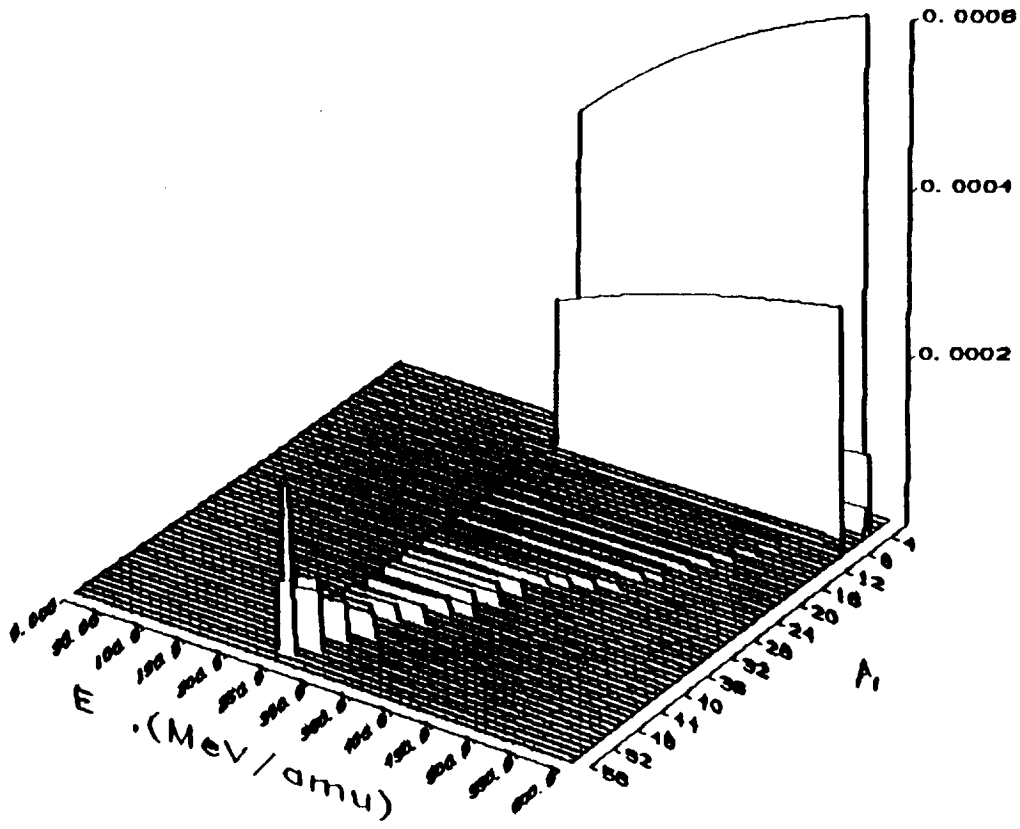

Figure 2.C. 2-nd term nonperturbation solution at a depth of 10 cm of water for a 600 MeV/nucleon Iron projectile.

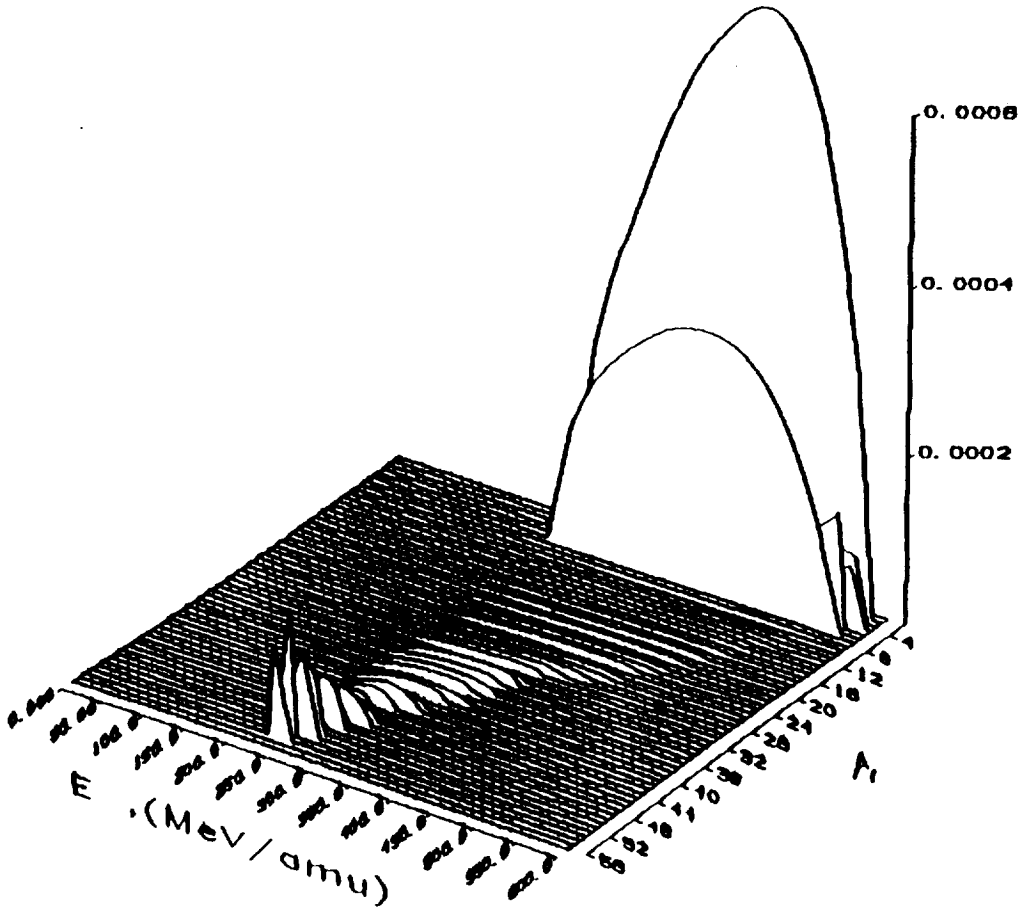

Figure 2.D. 2-nd term perturbation solution at a depth of 10 cm of water for a 600 MeV/nucleon Iron projectile.

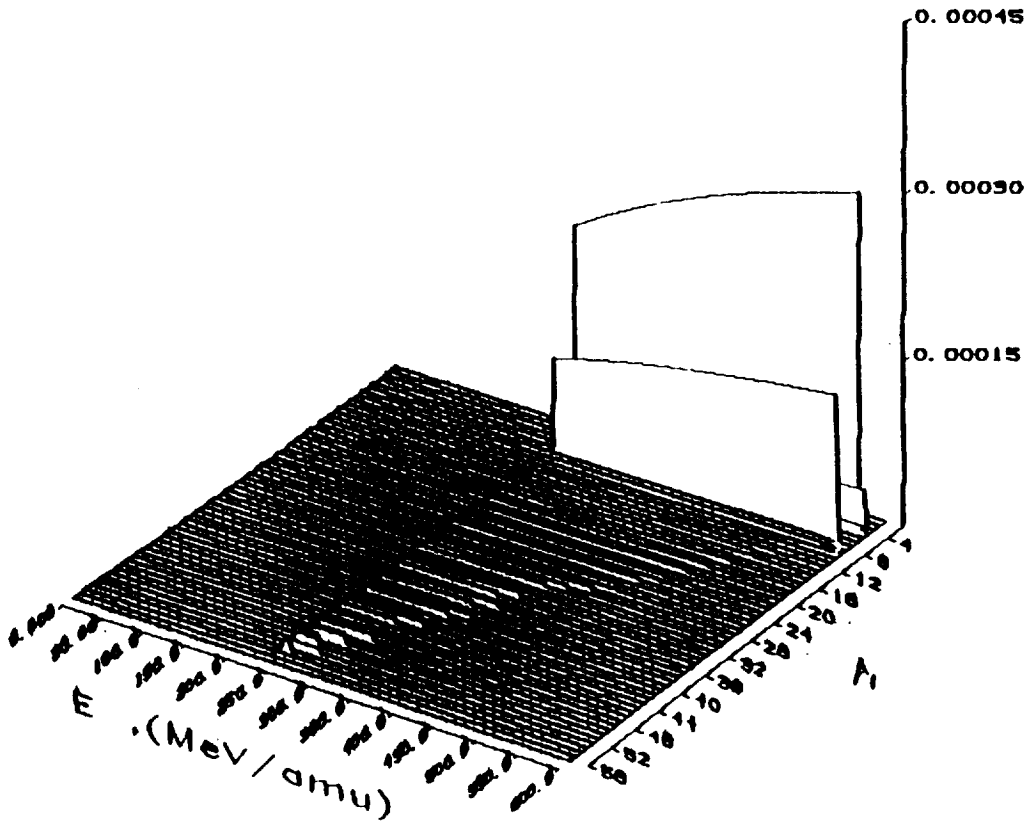

Figure 2.E. 3-rd term perturbation solution at a depth of 10 cm of water for a 600 MeV/nucleon Iron projectile.

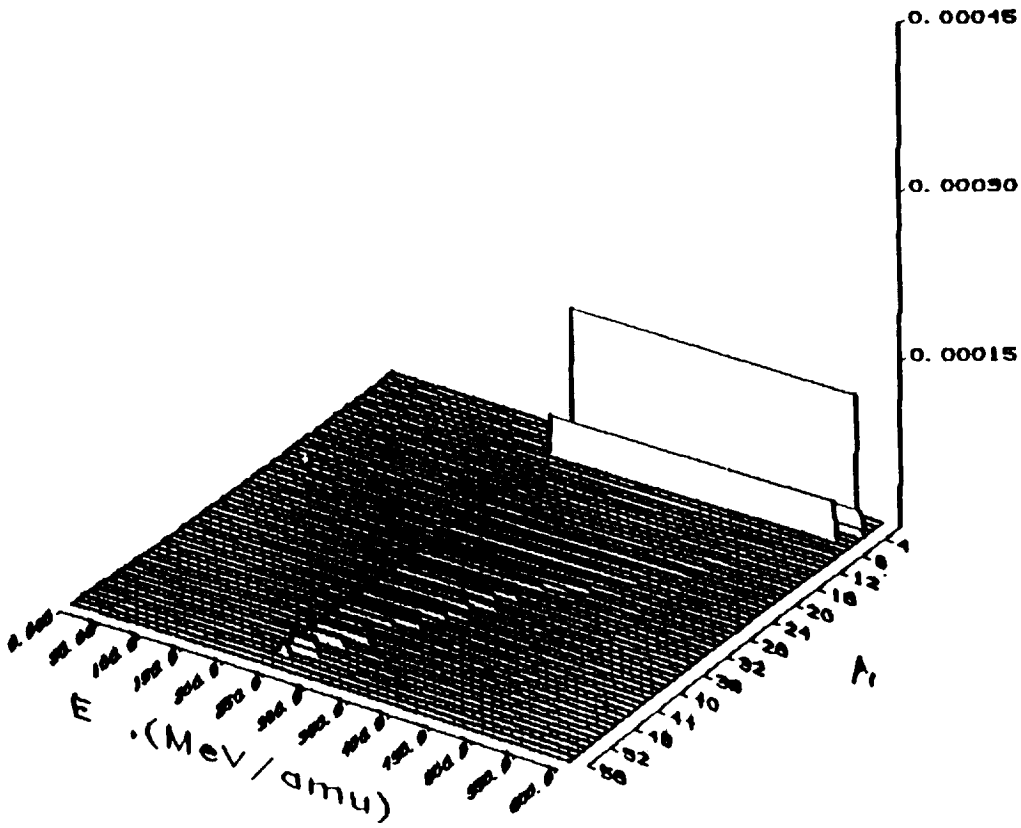

Figure 2.F. 3-rd term perturbation solution at a depth of 10 cm of water for a 600 MeV/nucleon Iron projectile.

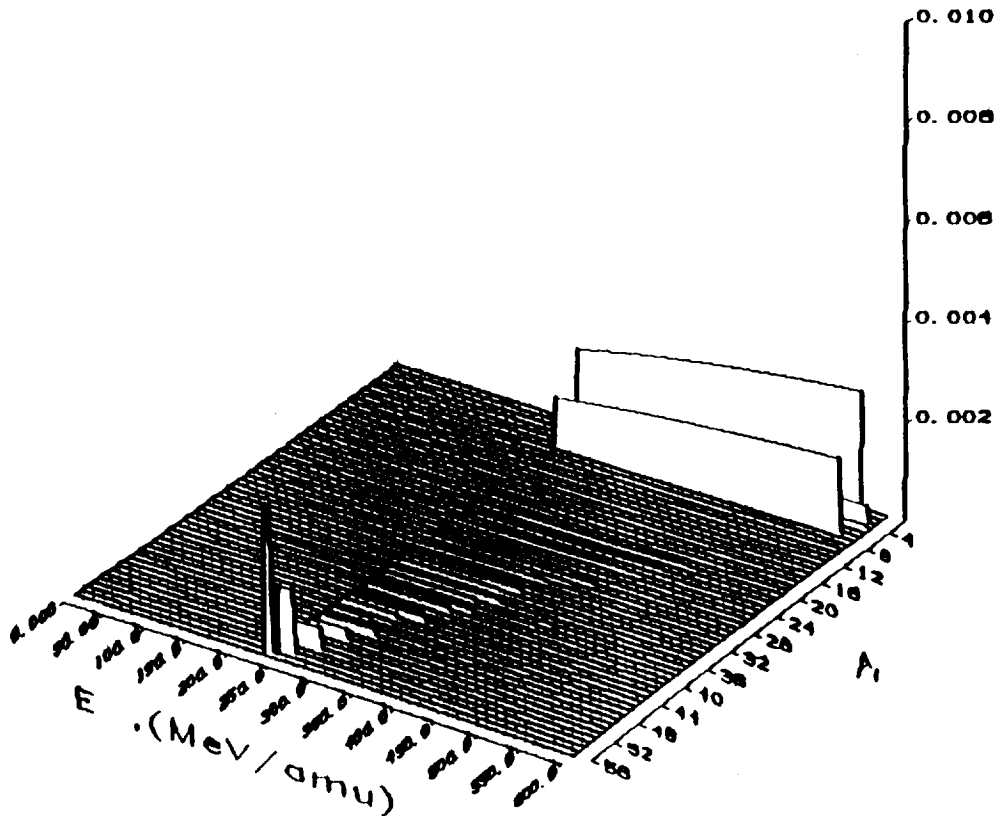

Figure 2.G. All terms nonperturbation solution at a depth of 10 cm of water for a 600 MeV/nucleon Iron projectile.

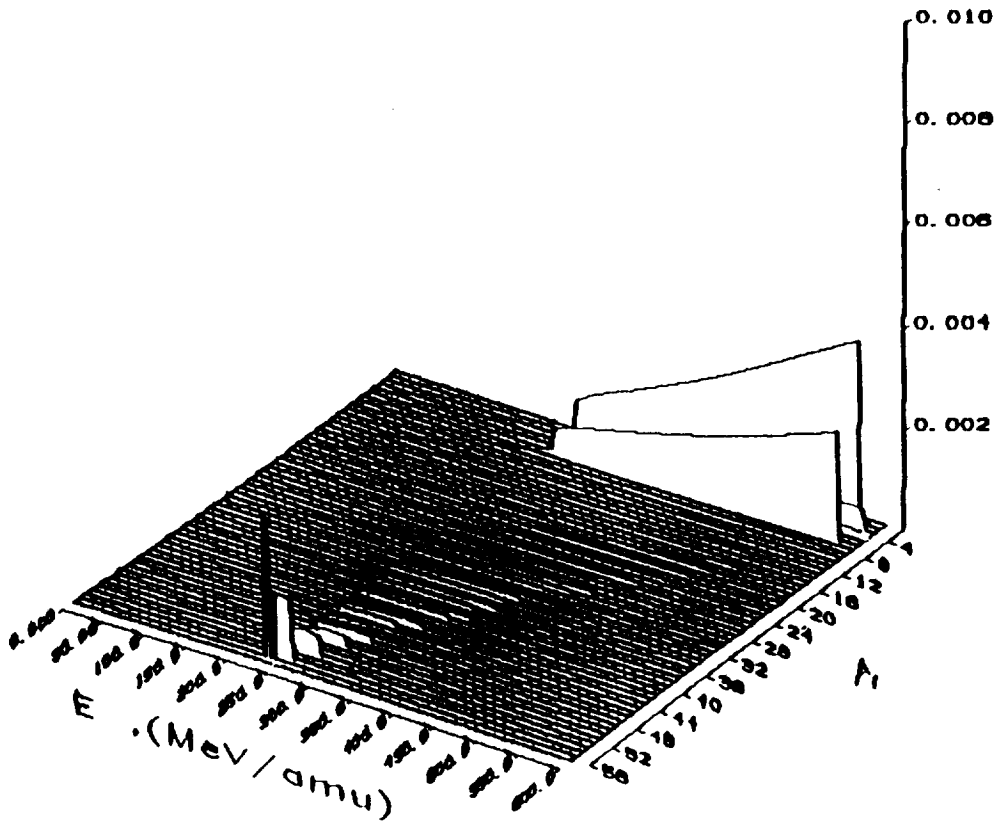

Figure 2.H. All terms perturbation solution at a depth of 10 cm of water for a 600 MeV/nucleon Iron projectile.

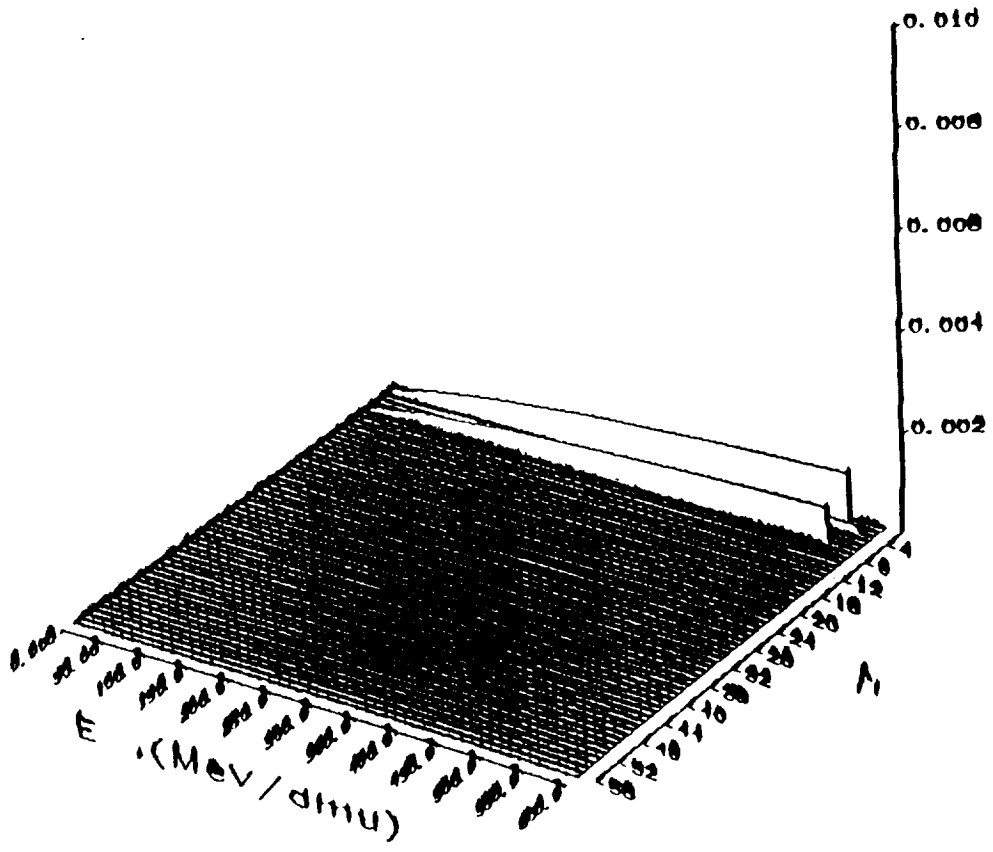

Figure 3.A. 1-st term nonperturbation solution for a depth of 15 cm of water for a600 MeV/nucleon Iron projectile.

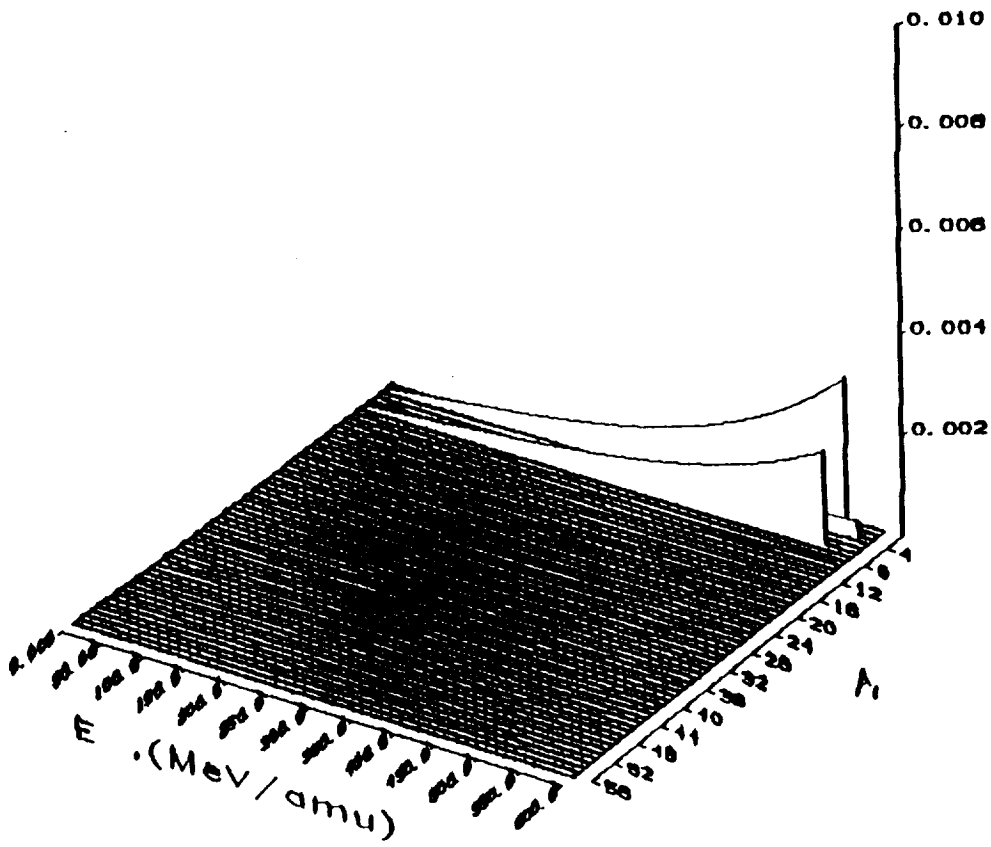

Figure 3.B. 1-st term perturbation solution for a depth of 15 cm of water for a 600 MeV/nucleon Iron projectile.

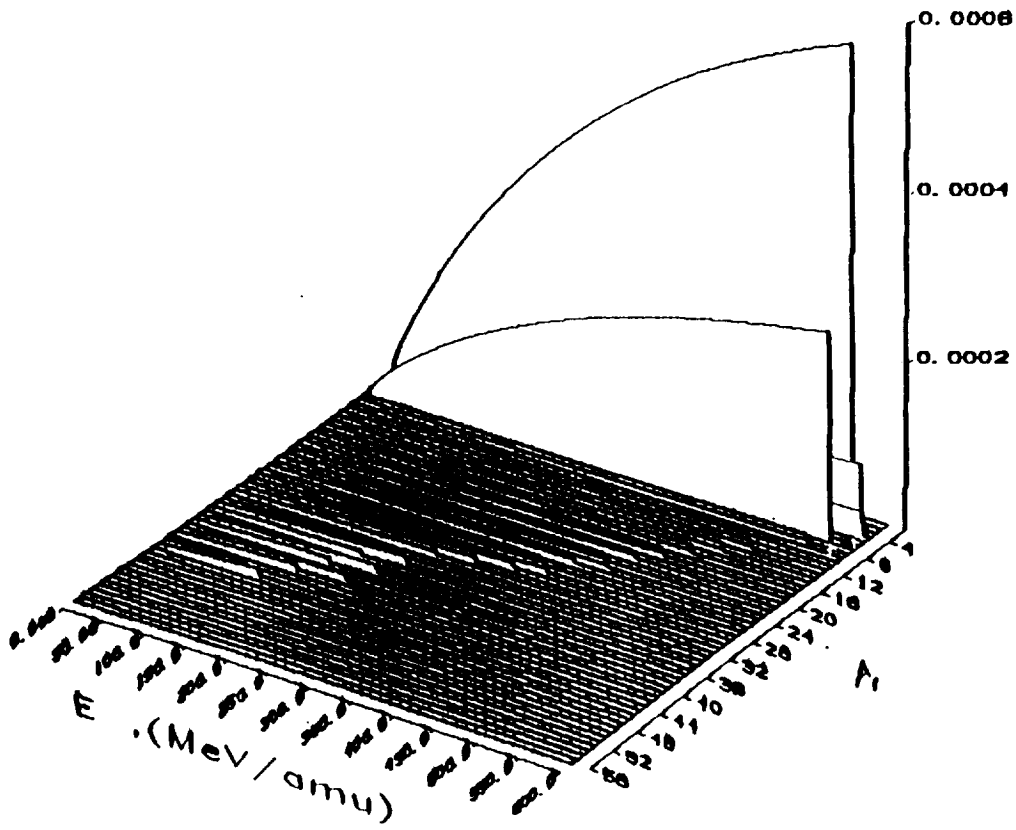

Figure 3.C. 2-nd term nonperturbation solution for a depth of 15 cm of water for a 600 MeV/nucleon Iron projectile.

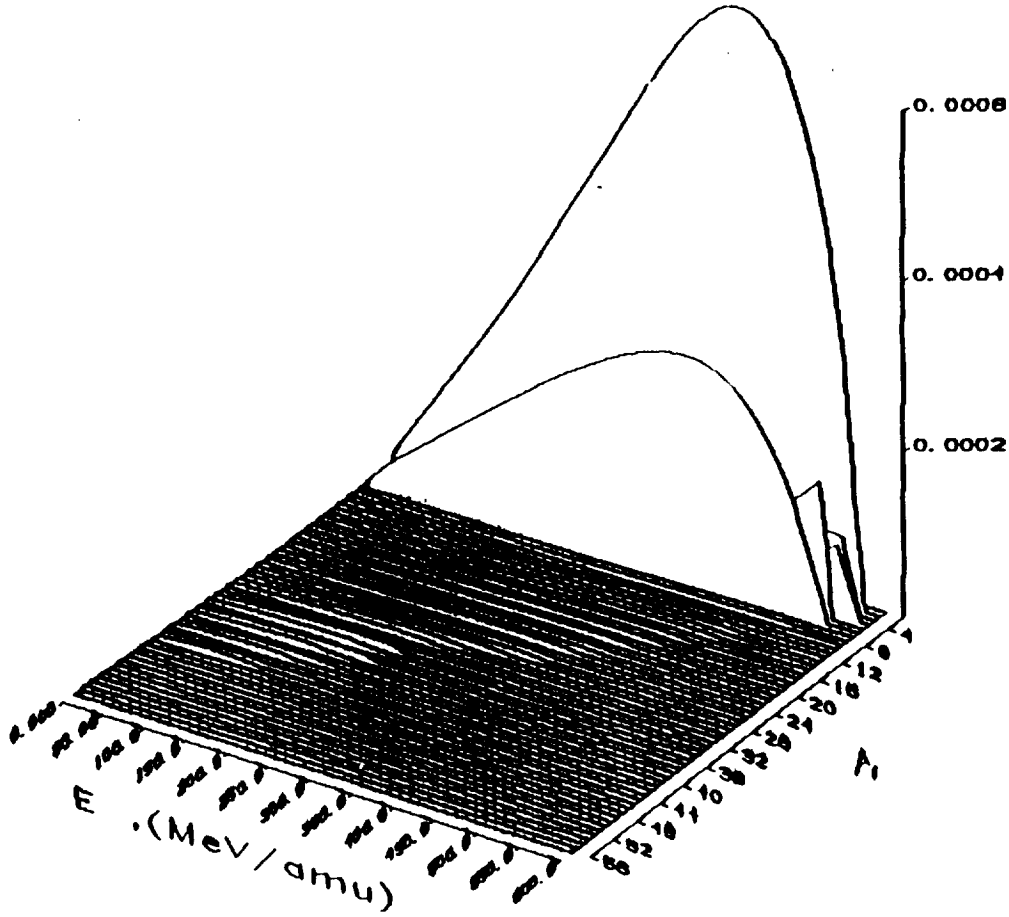

Figure 3.D. 2-nd term perturbation solution for a depth of 15 cm of water for a 600 MeV/nucleon Iron projectile.

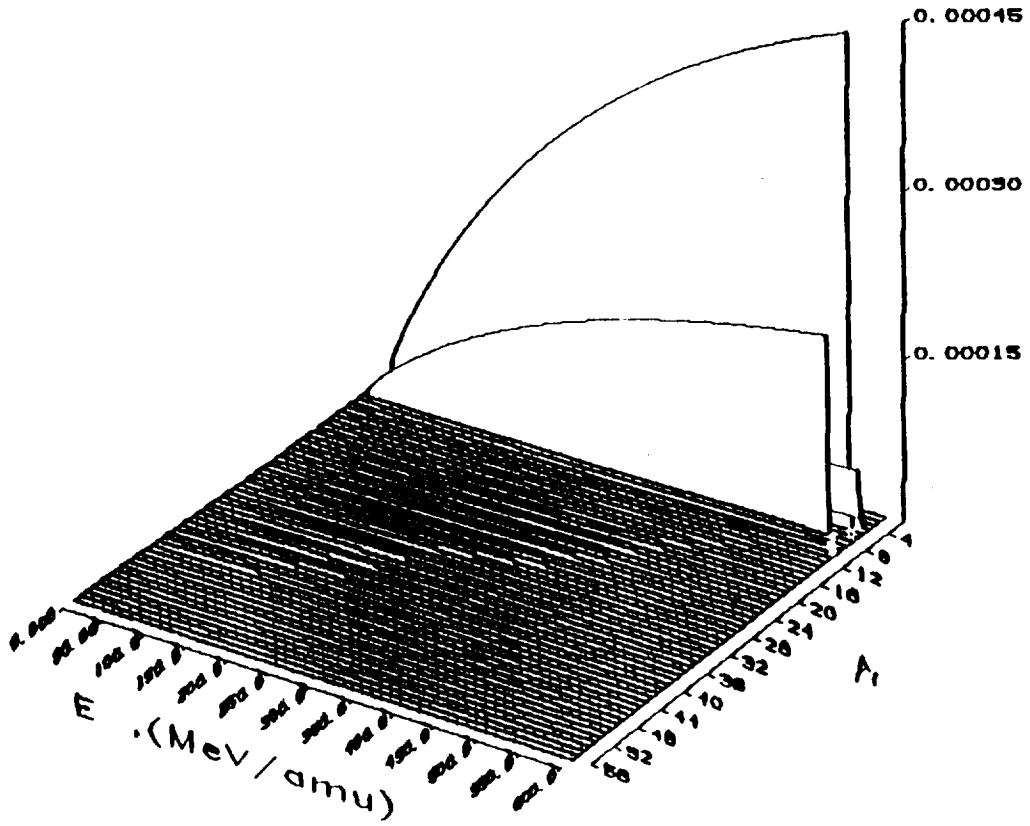

Figure 3.E. 3-rd term nonperturbation solution for a depth of 15 cm of water for a 600 MeV/nucleon Iron projectile.

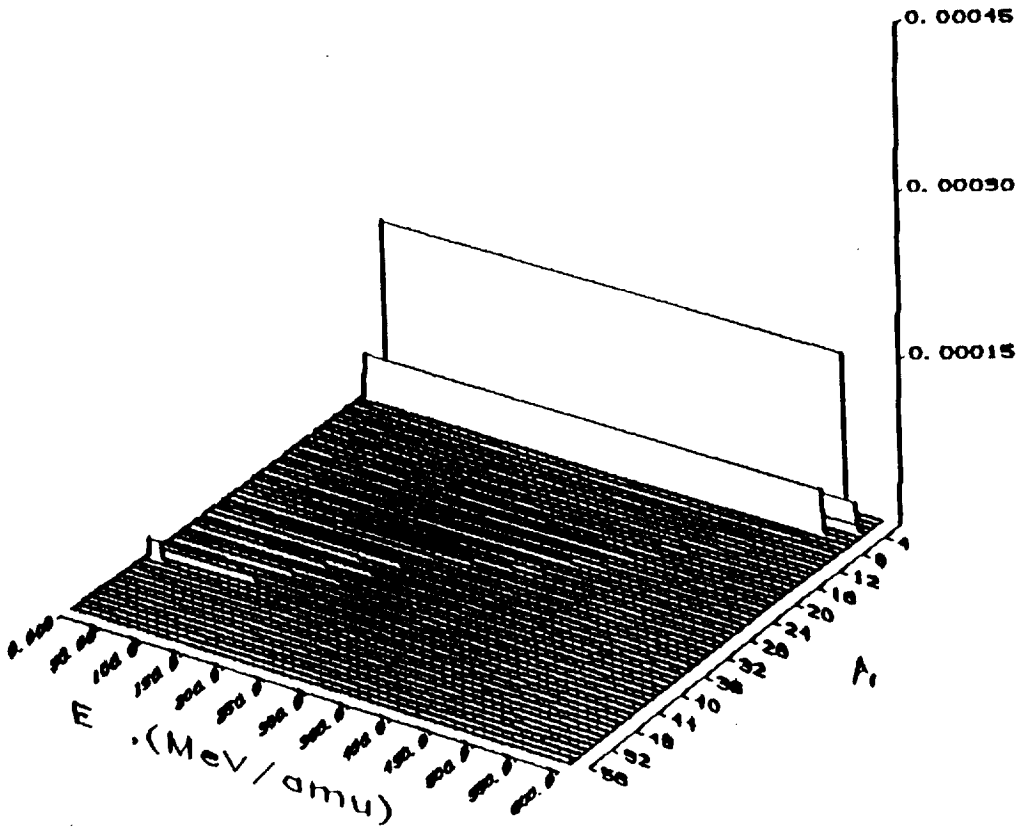

Figure 3.F. 3-rd term perturbation solution for a depth of 15 cm of water for a 600 MeV/nucleon Iron projectile.

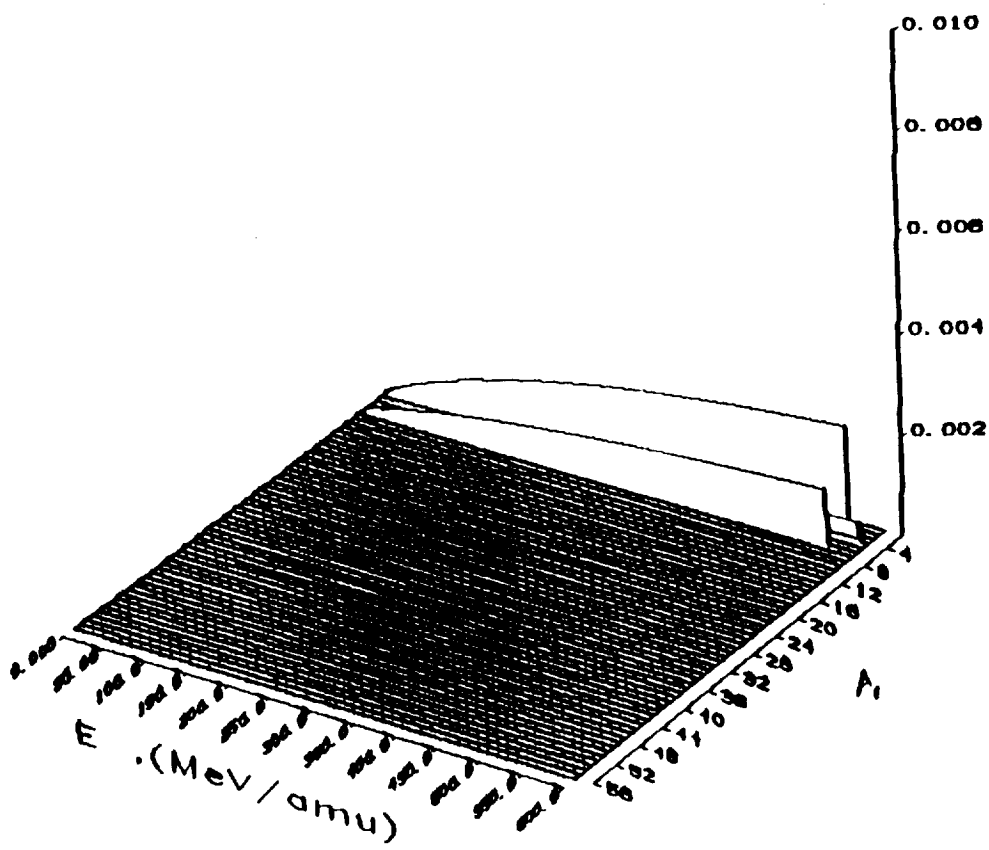

Figure 3.G. All terms nonperturbation solution for a depth of 15 cm of water for a 600 MeV/nucleon Iron projectile.

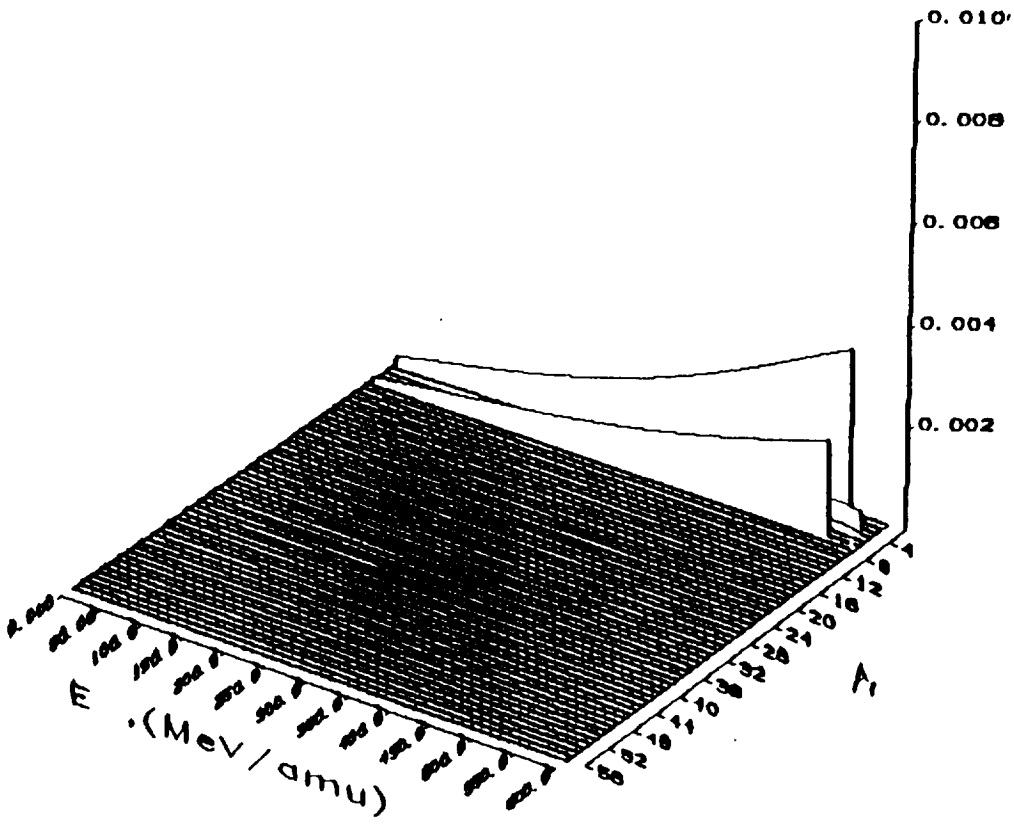

Figure 3.H. All terms perturbation solution for a depth of 15 cm of water for a 600 MeV/nucleon Iron projectile.

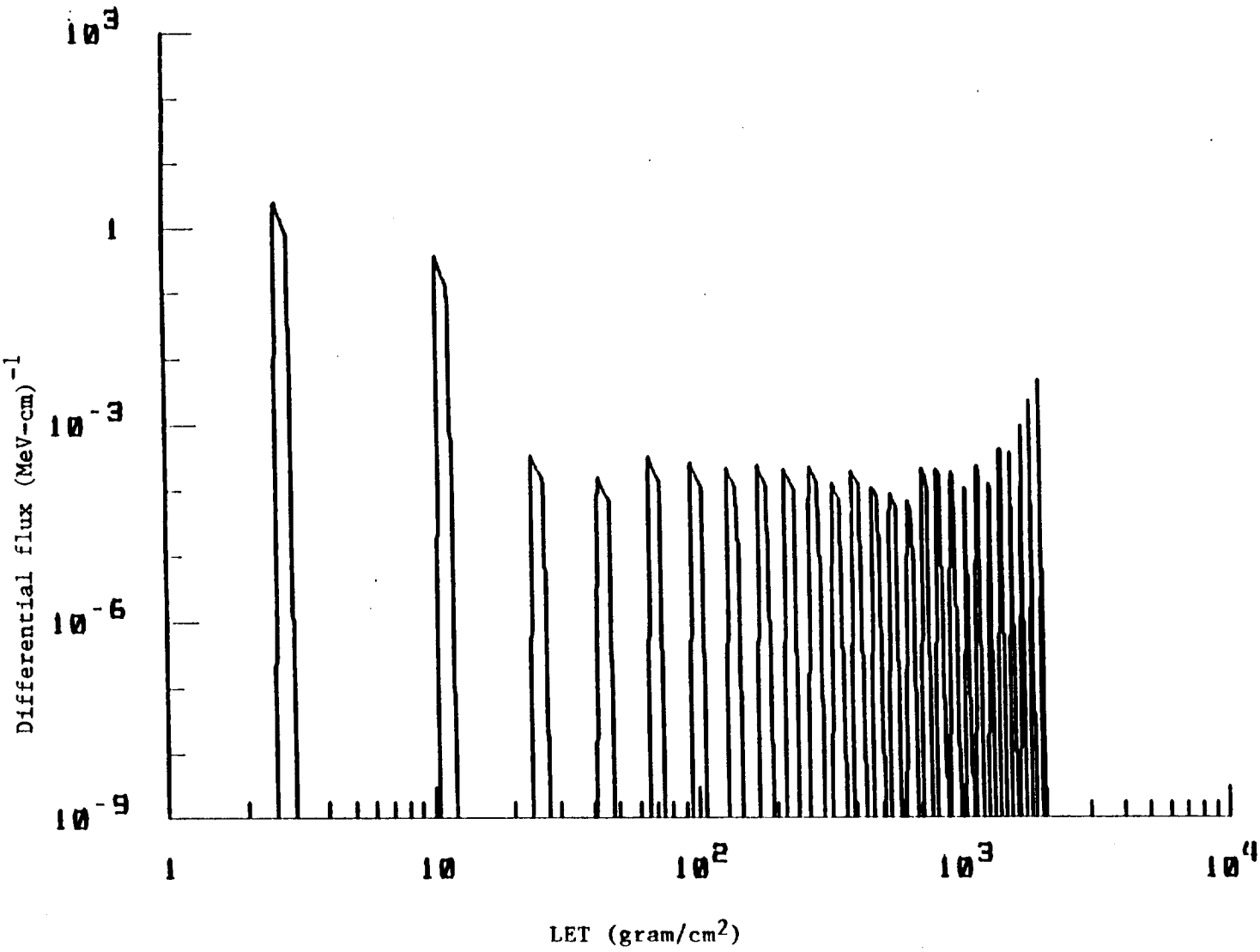

Figure 4.A. Differential LET spectrum for a depth of 5 cm of water for a 600 MeV/nucleon Iron projectile.

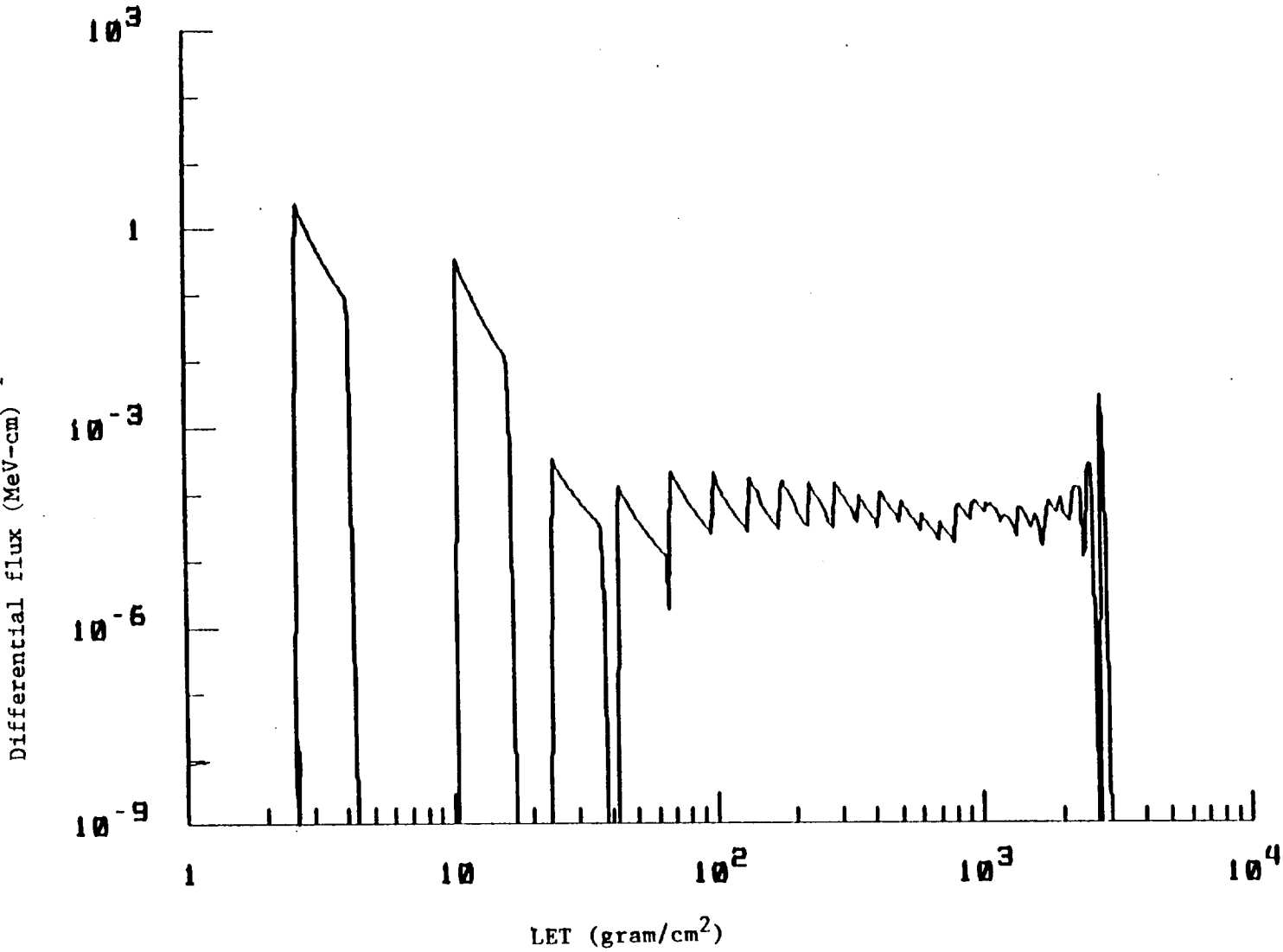

Figure 4.B. Differential LET spectrum for a depth of 10 cm of water for a 600 MeV/nucleon Iron projectile.

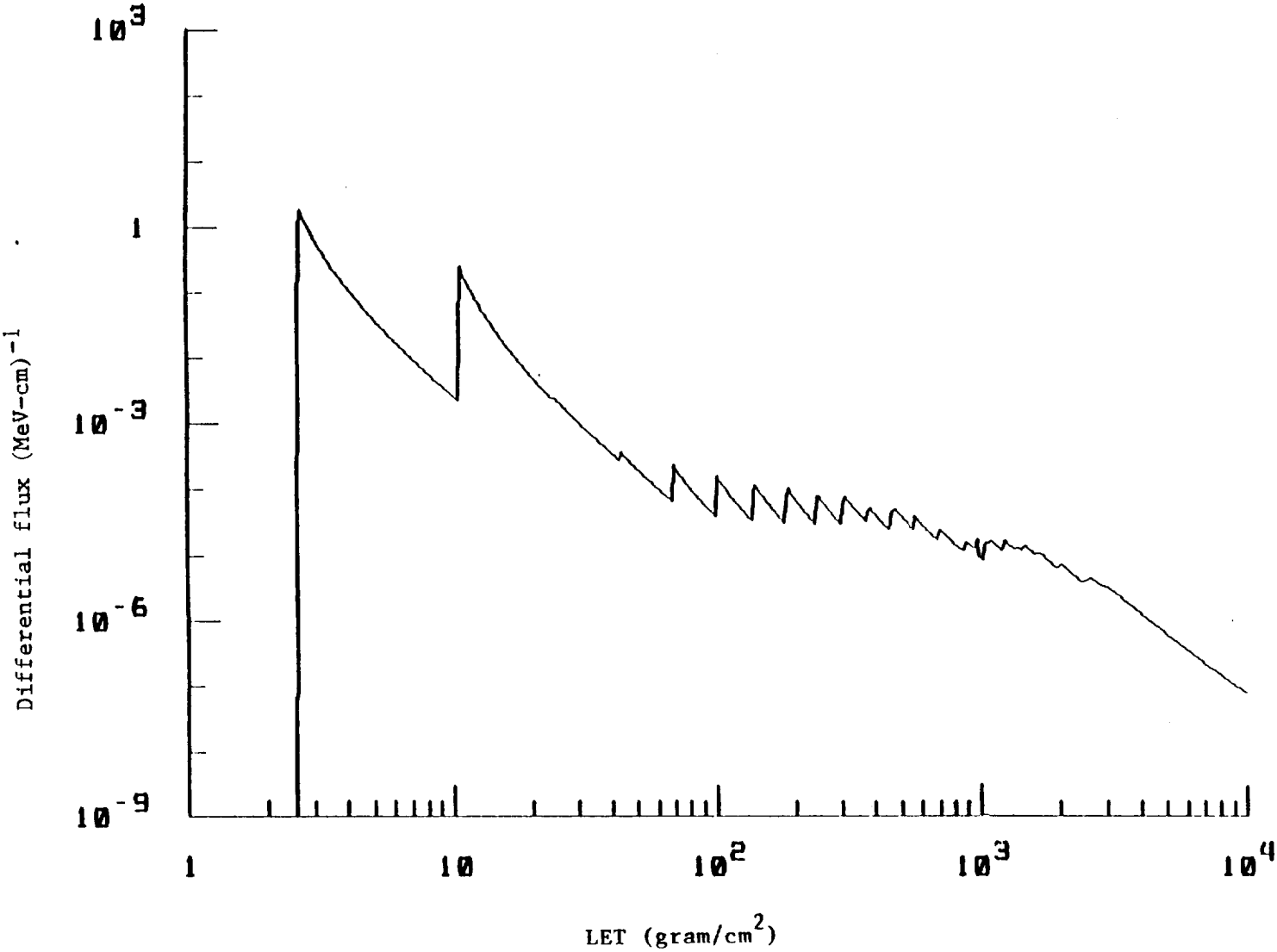

Figure 4.C. Differential LET spectrum for a depth of 15 cm of water for a 600 MeV/nucleon Iron projectile.

```
      PROGRAM GREEN_FUNCTION
C
      PARAMETER(IJ=59,IK=IJ*IJ,IL=IJ+1,IERS=301)
C
      COMMON/TARGET/NLAY,XLAY,DN,NAT,ATRG(5),ZTRG(5),DENSTRG(5)
C
      DIMENSION E0(IJ),G0(IJ,IJ),G1(IJ,IJ),G2(IJ,IJ),GG(IJ,IJ)
      DIMENSION EIL(IJ,IJ),EIU(IJ,IJ),EML(IJ,IJ),EMU(IJ,IJ)
      DIMENSION B(IJ,IJ),AT(IJ),ZT(IJ)
      DIMENSION ACC(0:IJ),PHI(IJ),ET(IERS),ST(IERS),RT(IERS),F(IERS,IJ)
C
      DATA E0,EIL,EIU,EML,EMU,ACC/IJ*0.,IK*0.,IK*0.,IK*0.,IK*0.,IL*0./
C
      NLAY=1
      NAT=2
      DN=1.
      ATRG(1)=1.
      ZTRG(1)=1.
      DENSTRG(1)=6.68E22
      ATRG(2)=16.
      ZTRG(2)=8.
      DENSTRG(2)=3.34E22
C
C   INITIALIZE TS ARRAY WITH A DUMMY CALL, SO A(J),Z(J),ANU(J)
C   ROUTINES ARE ACTIVATED
C
      DUMMY=TS(1.,1,1)
C
      OPEN(UNIT=2,FILE='FLUX.DAT',STATUS='NEW')
      OPEN(UNIT=3,FILE='GLUX.DAT',STATUS='NEW')
C
      X=5.
      EMAX=600.
C
C   PROJECTILE IS IRON 26,56 WITH AN INDEX OF 57 IN OUR ISOTOPE TABLE
C
      INDEXAP=57
C
      DO 3 II=1,IJ
      ZT(II)=Z(II)
    3 AT(II)=A(II)
C
      KOUNT=0
C
      DO 1 E=0.,EMAX,2.
      KOUNT=KOUNT+1
      EA=E
      IF(EA.LT.1.E-3) EA=1.E-3
      ET(KOUNT)=EA
      ST(KOUNT)=SMAT(EA,1.,1.)
      RT(KOUNT)=RMAT(EA,1.,1.)
      CALL FLUX(PHI,EA,X,INDEXAP)
C
      DO 2 I=1,IJ
      F(KOUNT,I)=PHI(I)*ST(KOUNT)*ANU(I)
    2 CONTINUE
C
    1 CONTINUE
C
      IF(KOUNT.NE.IERS) THEN
      PRINT *,' VALUE OF IERS IN PARAMETER STATEMENT IS INCORRECT'
      STOP
```

```
      END IF
      WRITE(2,*)IJ,AT,ZT
      WRITE(2,*)ET,RT,ST,F
C
C   WRITE OUT GRAPHICAL OUTPUT (3D), BUT DESCALE FLUX BY LET
C
      DO 5 K=1,KOUNT
C
      DO 5 I=1,INDEXAP
      WRITE(3,*)I,F(K,I)/(ST(K)*ANU(I)),ET(K)
    5 CONTINUE
C
      CLOSE(UNIT=2)
      CLOSE(UNIT=3)
      END
C
C*******************************************************
C
      SUBROUTINE FLUX(PHI,E,X,IP)
C
      PARAMETER(IJ=59,IK=IJ*IJ,IL=IJ+1)
C
      DIMENSION E0(IJ),G0(IJ,IJ),G1(IJ,IJ),G2(IJ,IJ),GG(IJ,IJ)
      DIMENSION EIL(IJ,IJ),EIU(IJ,IJ),EML(IJ,IJ),EMU(IJ,IJ)
      DIMENSION B(IJ,IJ),PHI(IJ)
C
      CALL GREEN(X,E,G0,G1,B,G2,GG,E0,EIL,EIU,EML,EMU)
C
      DO 1 I=1,IJ
      PHI(I)=0.
    1 CONTINUE
C
      CALL F0(E0(IP),IP,PSI,CPSI,EPSI)
C      IF(E0(IP).GT.500.)
C     -PRINT*,'E0,G0,PSI,IP=',E0(IP),G0(IP,IP),PSI,IP
      PHI(IP)=G0(IP,IP)*PSI
      IF(PHI(IP).LT.1.E-20)PHI(IP)=0.
C
C  ALL BUT NEUTRONS ARE CALCULATED FOR NOW (IE,J>1)
C
      DO 2 J=2,IP-1
      CALL F0(EIL(J,IP),IP,PSI,CPSIL,EPSIL)
      CALL F0(EIU(J,IP),IP,PSI,CPSIU,EPSIU)
      DNOM=CPSIL-CPSIU
      EAVE=(EIL(J,IP)+EIU(J,IP))/2.
      RAVE=RMAT(EAVE,A(IP),Z(IP))
      EAV=EAVE
      IF(DNOM.GT.0.) EAV=(EPSIL-EPSIU)/(CPSIL-CPSIU)
      IF(EAV.LT.EIL(J,IP))EAV=EIL(J,IP)
      IF(EAV.GT.EIU(J,IP))EAV=EIU(J,IP)
      IF((EAV-EIL(J,IP))*(EIU(J,IP)-EAV).LT.0.)
     -PRINT *,' EAV',EAV,EIL(J,IP),EIU(J,IP)
      RAV=RMAT(EAV,A(IP),Z(IP))
      PHI(J)=PHI(J)+(G1(J,IP)+B(J,IP)*
     -(RAV-RAVE))*(CPSIL-CPSIU)
C       IF(B(J,IP).GT.0.) PRINT *,' B=',B(J,IP),J,IP
      CALL F0(EML(J,IP),IP,PSI,CPSIL,EPSIL)
      CALL F0(EMU(J,IP),IP,PSI,CPSIU,EPSIU)
      PHI(J)=PHI(J)+(G2(J,IP)+GG(J,IP))*(CPSIL-CPSIU)
      IF(PHI(J).LT.1.E-20) PHI(J)=0.
    2 CONTINUE
C
      RETURN
      END
C
C *******************************************************
```

```
C
      SUBROUTINE FO(E,IP,PSI,CPSI,EPSI)
C
      PARAMETER(IJ=59,IK=IJ*IJ,IL=IJ+1)
C
C     THIS SUBROUTINE GENERATES THE SCALED FLUX AT THE BOUNDARY
C     AND THE CORRESPONDING INTEGRAL VALUES FOR LABORATORY SPECTRA
C
      DATA ZP,AP,E0,SIG,SQ2PI/26.,56.,600.,1.5,2.5066/
C
C     ZP,AP,E0,SIG DEFINES THE BEAM
C     ZP=CHARGE
C     AP=MASS
C     E0=MEAN ENERGY (MEV/AMU)
C     SIG=STANDARD DEVIATION OF ENERGY
C
C     IBEAM=1 GAUSSIAN DIST
C     IBEAM=0 DELTA DIST
C
      IBEAM=1
      IF(IBEAM.EQ.1) GO TO 1
      PSI=0.
      IF(E.GT.E0) THEN
      CPSI=0.
      EPSI=0.
      ELSE
      CPSI=1.
      EPSI=E0
      END IF
      RETURN
C
    1 PSI=SMAT(E,A(IP),Z(IP))*TEXP(-((E0-E)/SIG)**2/2.)
     -/(SQ2PI*SIG*A(IP))
      CPSI=(1.-ERF((E-E0)/(1.414*SIG)))/2.
      EPSI=E0*CPSI+SIG*TEXP(-((E-E0)/SIG)**2/2.)/SQ2PI
      RETURN
      END
C
C ***************************************************
C
      FUNCTION ERF(X)
C
      REAL*8 ERF,X
      REAL*8 PERF,PX
C
      PX= ABS(X)
      ERF=PERF(PX)
      IF(X.LT.0.)ERF=-ERF
      RETURN
      END
C
C ***************************************************
C
      FUNCTION PERF(X)
C
      REAL*8 PERF,X
      REAL*16 QX,QERF,T,A1,A2,A3
C
      DATA A1,A2,A3/.3480242D0,-.0958798D0,.7478556D0/
      QX=X
      T=1.D0/(1.D0+.47047D0*QX)
      QERF=T*(A1+T*(A2+T*A3))
      PERF=1.D0-QERF* EXP(-QX*QX)
      RETURN
      END
C
```

```
C ****************************************************
C
      FUNCTION GPER01(E0,IAP,E,IAF,X)
C
C    FUNCTION GPER01 IS FOR MONO E BEAM AND VELOCITY CONSERVING INTERACTIONS
C
      REAL NUP,NUF
C
      PSI01=0.
      GPER01=0.
      IF(E.LE.0.) RETURN
      IF(Z(IAF).LT.1.) RETURN
      FRAGZ=TS(E0,IAF,IAP)
      AP=A(IAP)
      AF=A(IAF)
      ZP=Z(IAP)
      ZF=Z(IAF)
      NUP=ZP*ZP/AP
      NUF=ZF*ZF/AF
      R0=RMAT(E0,AP,ZP)
      RJ=RMAT(E,AF,ZF)
      RJL=NUP*(R0-X)/NUF
      RJU=NUP*R0/NUF-X
      EJL=EMAT(RJL,AF,ZF)
      EJU=EMAT(RJU,AF,ZF)
      ETA=(2.*NUP*R0-(NUP+NUF)*(X+RJ))/(NUP-NUF)
      XF=(X-RJ-ETA)/2.
      XP=(X+RJ+ETA)/2.
      SIGF=XS(E0,IAF)-TS(E0,IAF,IAF)
      SIGP=XS(E0,IAP)-TS(E0,IAP,IAP)
      IF(X.EQ.0.)PRINT *,' AF,ZF,SIGP,SIGF,FRAG= ',AF,ZF,SIGP,SIGF,FRAGZ
      IF(XF.LT.0..OR.XP.LT.0.) GO TO 98
      PSI01=FRAGZ*NUF*TEXP(-SIGF*XF-SIGP*XP)/ABS(NUP-NUF)
      GPER01=PSI01/(SMAT(E,AF,ZF)/AF)
   98 CONTINUE
      RETURN
      END
C
C ***********************************************
C
      FUNCTION GPER02(E0,IAP,E,IAF,X)
C
C   FUNCTION GPER02 EVALUATES THE SECOND TERM IN PERTURBATION
C   SERIES. THE SOLUTION IS ENERGY INDEPENDENT X SECTIONS
C   AND IGNORES THE MOMENTUM SPREAD CAUSED BY FRAGMENTATION.
C   IE, VELOCITY CONSERVING INTERACTIONS.  ALSO A MONOENERGETIC BEAM.
C
      REAL NUP,NUJ,NUK

      DATA ZERO/-1.E-3/
C
      GPER02=0.
      IF(IAF.GE.IAP-1) RETURN
      IF(Z(IAF).LT.1.) RETURN
      AP=A(IAP)
      ZP=Z(IAP)
      NUP=ANU(IAP)
      SIGP=XS(E0,IAP)-TS(E0,IAP,IAP)
      JAF=IAF
      AFJ=A(IAF)
      ZFJ=Z(IAF)
      NUJ=ANU(IAF)
      SIGJ=XS(E0,IAF)-TS(E0,IAF,IAF)
C
      DO 99 K=IAF+1,IAP-1
      KAF=K
```

```
      AFK=A(KAF)
      ZFK=Z(KAF)
      IF(ZFK.LT.1.) GO TO 98
      NUK=ANU(KAF)
      SIGK=XS(EO,KAF)-TS(EO,KAF,KAF)
      FRAGK=TS(EO,KAF,IAP)
      FRAGJ=TS(EO,IAF,KAF)
      RO=RMAT(EO,AP,ZP)
      XJ1=(NUK*(RMAT(E,AFK,ZFK)+X)-NUP*RO)/(NUK-NUJ)
      XJ2=NUP*(RMAT(E,AP,ZP)+X-RO)/(NUP-NUJ)
      IF(NUK.GT.NUP) GO TO 66
      IF(NUJ.GT.NUK) GO TO 77
      XJL=0.
      IF(XJ1.GT.XJL) XJL=XJ1
      XJU=X
      IF(XJ2.LT.XJU) XJU=XJ2
      GO TO 80
   66 XJL=0.
      IF(XJL.LT.XJ2) XJL=XJ2
      XJU=X
      IF(XJU.GT.XJ1) XJU=XJ1
      GO TO 80
   77 XJL=0.
      XJU=X
      IF(XJU.GT. XJ1) XJU= XJ1
      IF(XJU.GT.XJ2) XJU=XJ2
   80 CONTINUE
      IF(XJU.LT.XJL) GO TO 96
      XML=(NUP*RO-NUK*(RMAT(E,AFK,ZFK)+X)+(NUK-NUJ)*XJL)
     -/(NUP-NUK)
      XKL=(NUP*(RMAT(E,AP,ZP)+X-RO)-(NUP-NUJ)*XJL)
     -/(NUP-NUK)
      XMU=(NUP*RO-NUK*(RMAT(E,AFK,ZFK)+X)+(NUK-NUJ)*XJU)
     -/(NUP-NUK)
      XKU=(NUP*(RMAT(E,AP,ZP)+X-RO)-(NUP-NUJ)*XJU)
     -/(NUP-NUK)
      FRAGJ=TS(EO,IAF,KAF)
      IF(XMU.LT.ZERO)GO TO 96
      IF(XKU.LT.ZERO) GO TO 96
      IF(XJU.LT.ZERO) GO TO 96
      IF(XML.LT.ZERO) GO TO 96
      IF(XKL.LT.ZERO) GO TO 96
      IF(XJL.LT.ZERO) GO TO 96
      ARGL=SIGP*XML+SIGK*XKL+SIGJ*XJL
      ARGU=SIGP*XMU+SIGK*XKU+SIGJ*XJU
      SLOPE=SIGJ+((NUK-NUJ)*SIGP-(NUP-NUJ)*SIGK)/(NUP-NUK)
      SLOPE=ABS(NUP-NUK)*SLOPE
      VALUE=(TEXP(-ARGL)-TEXP(-ARGU))/SLOPE
      GPER02=GPER02+FRAGJ*FRAGK*NUJ*VALUE
   96 CONTINUE
   97 CONTINUE
   98 CONTINUE
   99 CONTINUE
C
      GPER02=GPER02/(SMAT(E,AFJ,ZFJ)/AFJ)
      RETURN
      END
C
C ******************************************************
C
      FUNCTION GPER03(EP,IAP,E,IAS,X)
C
C     FUNCTION GPER03 IS AN APPROXIMATE EVALUATION OF THIRD COLLISION TERM
C     OF THE PERTURBATION SERIES. X-SECTIONS ARE CONSTANT AND THE BEAM IS
C     MONOENERGETIC. USE IT AS RECUIRED TIL SOME THING BETTER
C     COMES ALONG. PROBABLY NOT TOO BAD.
```

```
C
      REAL NUP,NUJ,NUK
C
      GLX3=0.
      IF(Z(IAS).LE.0.) RETURN
      IF(Z(IAP).LT.1.) RETURN
      ZP=Z(IAP)
      AP=A(IAP)
      ZS=Z(IAS)
      AS=Z(IAZ)
      NUP=ANU(IAP)
      NUJ=ANU(IAS)
      GPER03=0.
      IA1=IAS+1
      IA1MAX=IAP-1.
C
      DO 1 K1=IA1,IA1MAX
      AK1=A(K1)
      ZK1=Z(K1)
      SIGK1=XS(EP,K1)-TS(EP,K1,K1)
      SIGP=XS(EP,IAP)-TS(EP,IAP,IAP)
      SIGS=XS(EP,IAS)-TS(EP,IAS,IAS)
      FRAGK1=TS(EP,K1,IAP)
      IA2=IAS+1
      IA2MAX=K1-1
C
      DO 3 K2=IA2,IA2MAX
      AK2=A(K2)
      ZK2=Z(K2)
      NUK=ANU(K2)
      SIGK2=XS(EP,K2)-TS(EP,K2,K2)
      FRAGK2=TS(EP,K2,K1)
      FRAGS=TS(EP,IAS,K2)
      RO=RMAT(EP,AP,ZP)
      RLJ=RO-X
      RUJ=(NUP*RO-NUJ*X)/NUK
      EUJ=EMAT(RUJ,AK2,ZK2)
      ELJ=EMAT(RLJ,AP,ZP)
      IF(NUP.GT.NUK) GO TO 7
      RLJ=NUP*RO/NUK-X
      RUJ=RO-NUJ*X/NUP
      ELJ=EMAT(RLJ,AK2,ZK2)
      EUJ=EMAT(RUJ,AP,ZP)
      IF(NUK.GT.NUP) GO TO 7
      RLJ=RO-X
      RUJ=NUP*RO/NUK-X
      ELJ=EMAT(RLJ,AP,ZP)
      EUJ=EMAT(RUJ,AK2,ZK2)
    7 CONTINUE
      IF(E.GT.EUJ.OR.E.LT.ELJ) GO TO 3
      GLX3=GLX3+FRAGS*FRAGK2*FRAGK1
     -*G3(SIGS,SIGK2,SIGK1,SIGP,X)/(EUJ-ELJ)
    3 CONTINUE
C
    2 CONTINUE
    1 CONTINUE
C
      GPER03=GLX3
      RETURN
      END
C
C ******************************************************
C
      FUNCTION G1(A,B,X)
C
      BB=B
```

```
      IF(A.EQ.B) BB=1.0000001*B
      G1=(TEXP(-A*X)-TEXP(-BB*X))/(BB-A)
      RETURN
      END
C
C *******************************************************
C
      FUNCTION G2(A,B,C,X)
C
      CC=C
      IF(B.EQ.C) CC=1.000001*C
      BB=B
      IF(A.EQ.B)BB=1.0000005*B
      G2=(G1(A,BB,X)-G1(A,CC,X))/(CC-BB)
      RETURN
      END
C
C *******************************************************
C
      FUNCTION G3(A,B,C,D,X)
C
      DD=D
      IF(C.EQ.D) DD=1.00001*D
      CC=C
      IF(B.EQ.C) CC=1.000005*C
      G3=(G2(A,B,CC,X)-G2(A,B,DD,X))/(DD-CC)
      RETURN
      END
C
C *******************************************************
C
      FUNCTION G4(A,B,C,D,E,X)
C
      EE=E
      IF(D.EQ.E) EE=1.0001*E
      G4=(G3(A,B,C,D,X)-G3(A,B,C,EE,X))/(EE-D)
      RETURN
      END
C
C *******************************************************
C
      FUNCTION TS(E,J,K)
C
      PARAMETER (NE=7,IJ=59,NUM1=NE*IJ,NUM2=NUM1*IJ)
C
      DIMENSION TST(NE,IJ,IJ),XST(NE,IJ),AA(5),ZZ(5),DD(5),ET(NE)
      DIMENSION ATST(IJ),ZTST(IJ)
C
      COMMON/TARGET/NLAY,XLAY,DN,NAT,ATRG(5),ZTRG(5),DENSTRG(5)
C
      DATA NOLD/0/
      DATA ET/25.,75.,150.,300.,600.,1200.,2400./
      DATA TST/NUM2*0./
      DATA XST/NUM1*0./
C
      NE1=NE
      N=1
      IF(NLAY.NE.NOLD) GO TO 80
   30 IE=1
   38 IF(IE.GT.NE1) GO TO 35
      ET1=ET(IE)
      IF(E.LT.ET1) GO TO 39
      IE=IE+1
      GO TO 38
   39 IF(IE.EQ.1) GO TO 35
      TS1=TST(IE,J,K)
```

```
      TS2=TST(IE-1,J,K)
      TS=TS2+(TS1-TS2)*(E-ET(IE-1))/(ET(IE)-ET(IE-1))
      GO TO 36
   35 IF(IE.EQ.1) TS=TST(IE,J,K)
      IF(IE.GT.NE1) TS=TST(NE1,J,K)
   36 RETURN
C
      ENTRY XS(E,J)
      N=2
      IF(NLAY.NE.NOLD) GO TO 80
   40 IE=1
   43 IF(E.LT.ET(IE).OR.IE.GT.NE1) GO TO 44
      IE=IE+1
      GO TO 43
   44 IF(IE.EQ.1.OR.IE.GT.NE1) GO TO 45
      XS1=XST(IE,J)
      XS2=XST(IE-1,J)
      XS=XS2+(XS1-XS2)*(E-ET(IE-1))/(ET(IE)-ET(IE-1))
      RETURN

   45 IF(IE.EQ.1) XS=XST(IE,J)
      IF(IE.GT.NE1) XS=XST(IE-1,J)
      RETURN
C
      ENTRY ANU(J)
      ANU=ZTST(J)**2/ATST(J)
      IF(ATST(J).EQ.4.) ANU=1.0001
      IF(ZTST(J).EQ.0.) ANU=.9999
      RETURN
C
      ENTRY A(J)
      A=ATST(J)
      RETURN
C
      ENTRY Z(J)
      Z=ZTST(J)
      RETURN
C
   80 CONTINUE
   90 OPEN(100,FILE='NEWNUC59.DAT',STATUS='OLD')
      REWIND(100)
  997 CONTINUE
      READ(100,*,END=994) IJTST,ATST,ZTST
C 9999 FORMAT(I5/4(8F8.2/)/4(8F8.2/))
C      PRINT 9999,IJTST,ATST,ZTST
      READ(100,*,END=994)NN,AA,ZZ,DD
      READ(100,*,END=994) ET,XST,TST
  990 FORMAT(I5/5F10.3/5F10.1/5E12.3/7F10.1/870(7E13.3/))
      PRINT*,' TS FOR ',NN,' ELEMENTS OF CHARGE ',(ZZ(I),I=1,NN)
      IF(IJ.NE.IJTST) GO TO 997
      IF(NN.NE.NAT) GO TO 997
C
      DO 995 JJN=1,NN
      IF(AA(JJN).NE.ATRG(JJN)) GO TO 997
      IF(ZZ(JJN).NE.ZTRG(JJN)) GO TO 997
      IF(DD(JJN).NE.DENSTRG(JJN)) GO TO 997
  995 CONTINUE
C
      PRINT *,' MATERIAL FOR TS ',NLAY,' FOUND ON NUCLEAR FILE '
     -,' THERE ARE ',NAT,' ELEMENTS OF CHARGES ',(ZTRG(I),I=1,NAT)
      GO TO 996
  994 CONTINUE
      PRINT *,' MATERIAL FOR TS ',NLAY,' NOT FOUND ON NUCLEAR FILE '
      STOP
  996 CONTINUE
      CLOSE(100,STATUS='KEEP')
```

```
      NOLD=NLAY
      GO TO (30,40)N
      END
C
C ****************************************
C
      SUBROUTINE GREEN(X,E,G0,G1,B,G2,GM,E0,EIL,EIU,EML,EMU)
C
      PARAMETER (NE=7,IJ=59,NUM1=NE*IJ,NUM2=NUM1*IJ,IJM=IJ*IJ)
C
      DIMENSION E0(IJ),G0(IJ,IJ),G1(IJ,IJ),G2(IJ,IJ),GM(IJ,IJ)
      DIMENSION EIL(IJ,IJ),EIU(IJ,IJ),EML(IJ,IJ),EMU(IJ,IJ)
      DIMENSION B(IJ,IJ)
C
      CALL GNFR(X,G0,G1,B,G2,GM)
C
      IF(E.LT.1.E-10) E=1.E-10
C
      DO 1 M=1,IJ
      IF(Z(M).GT.0.) THEN
      E0(M)=X+RMAT(E,A(M),Z(M))
      E0(M)=EMAT(E0(M),A(M),Z(M))
      ELSE
      E0(M)=E
      END IF
C
      DO 1 J=1,M
      IF(Z(J).GT.0.) THEN
      EIL(J,M)=ANU(J)*(RMAT(E,A(J),Z(J))+X)/ANU(M)
      EIU(J,M)=X+ANU(J)*RMAT(E,A(J),Z(J))/ANU(M)
      ELSE
      EIL(J,M)=E
      EIU(J,M)=E
      END IF
      IF(Z(M).GT.0.) THEN
      EML(J,M)=EMAT(EIL(J,M),A(M),Z(M))
      EMU(J,M)=EMAT(EIU(J,M),A(M),Z(M))
      ELSE
      EML(J,M)=E
      EMU(J,M)=E
      END IF
      IF(EML(J,M).GT.EMU(J,M)) THEN
      AAAAB=EML(J,M)
      EML(J,M)=EMU(J,M)
      EMU(J,M)=AAAAB
      END IF
      EIL(J,M)=EML(J,M)
      EIU(J,M)=EMU(J,M)
      SJ=SMAT(E,A(J),Z(J))/A(J)
      IF (Z(J).EQ.0.)SJ=1.
      G0(J,M)=G0(J,M)/SJ
      G1(J,M)=G1(J,M)/SJ
      B(J,M)=B(J,M)/SJ
      G2(J,M)=G2(J,M)/SJ
      GM(J,M)=GM(J,M)/SJ
    1 CONTINUE
C
      RETURN
      END
C
C ****************************************
C
      SUBROUTINE GNFR(XA,G0,G1,B,G2,GG)
C
      PARAMETER(IJ=59,IJM=IJ*IJ)
```

```
C
      DIMENSION G0(IJ,IJ),G1(IJ,IJ),G2(IJ,IJ),GG(IJ,IJ)
      DIMENSION B(IJ,IJ)
C
      X=XA
      IF(XA.LT.1.E-5) X=1.E-5
      CALL GPERT(G0,G1,G2,X)
      CALL GNF(X,GG)
C
      DO 1 I=1,IJ
C
      DO 1 J=1,IJ
      GG(I,J)=GG(I,J)-G1(I,J)-G2(I,J)
      B(I,J)=0.
      IF(I.LT.J) THEN
      B(I,J)=ANU(I)*ANU(J)*TS(2000.,I,J)*(G0(J,J)-G0(I,I))
     1/((ANU(J)-ANU(I))*ABS(ANU(J)-ANU(I))*X)
      END IF
      IF (GG(I,J).LE.0.)GG(I,J)=0.
      IF(I.EQ.J) THEN
      GG(I,J)=0.
      GO TO 1
      END IF
      DN1=X*(ANU(J)/ANU(I)-1.)
      DN2=X*(ANU(J)/ANU(I)-1.)
      G1(I,J)=G1(I,J)/ABS(DN1)
      G2(I,J)=G2(I,J)/ABS(DN2)
      GG(I,J)=GG(I,J)/ABS(DN2)
      IF(XA.EQ.0.) THEN
      G2(I,J)=0.
      GG(I,J)=0.
      END IF
    1 CONTINUE
C
      RETURN
      END
C
C ***************************************************************
C
      SUBROUTINE GNF(X,G)
C
      PARAMETER(IJ=59,IL=12,IK=IL*IJ*IJ)
C
      COMMON/TARGET/NLAY,XLAY,DN,NAT,ATRG(5),ZTRG(5),DENSTRG(5)
C
      DIMENSION G(IJ,IJ),GT(IL,IJ,IJ),G0(IJ,IJ),G1(IJ,IJ),G2(IJ,IJ)
     1          ,XT(IL)
C
      DATA XT/0.,1.,2.,4.,6.,8.,14.,20.,34.,48.,62.,96./
      DATA NOLD,IPT,GT/0,-1,IK*0./
C
      E0=2000.
      IF (NLAY.NE.NOLD) GO TO 100
    2 CONTINUE
      NTAB=IJ**2
      CALL IUNI(IL,IL,XT,NTAB,GT,2,X,G,IPT,IER)
      IF(IER.NE.0.AND.IER.NE.-4) THEN
      PRINT *,' IUNI IN MODULE GNF FAILED WITH ERROR=',IER
      STOP
      END IF
C
      DO 1 I=1,IJ
C
      DO 1 J=1,IJ
      IF(I.EQ.J)G(I,J)=TEXP(G(I,J))
      IF(I.LT.J.AND.TS(E0,I,J).GT.0.)G(I,J)=X*TEXP(G(I,J))
```

```
    1 CONTINUE
C
      RETURN
  100 CONTINUE
C
      DO 101 I=1,IJ
C
      DO 101 J=1,IJ
      GT(1,I,J)=0.
      IF(I.EQ.J) GT(1,I,J)=1.
  101 CONTINUE
C
      NOLD=NLAY
      CALL GPERT(G0,G1,G2,XT(2)/2.)
C
      DO 1111 I=1,IJ
C
      DO 1111 J=1,IJ
      G0(I,J)=G0(I,J)+G1(I,J)+G2(I,J)
 1111 CONTINUE
C
      DO 102 I=1,IJ
C
      DO 102 J=1,IJ
      GT(2,I,J)=0.
C
      DO 102 K=1,IJ
      GT(2,I,J)=GT(2,I,J)+G0(I,K)*G0(K,J)
  102 CONTINUE
C
      DO 103 I=1,IJ
C
      DO 103 J=1,IJ
      GT(3,I,J)=0.
C
      DO 103 K=1,IJ
      GT(3,I,J)=GT(3,I,J)+GT(2,I,K)*GT(2,K,J)
  103 CONTINUE
C
      DO 104 I=1,IJ
C
      DO 104 J=1,IJ
      GT(4,I,J)=0.
C
      DO 104 K=1,IJ
      GT(4,I,J)=GT(4,I,J)+GT(3,I,K)*GT(3,K,J)
  104 CONTINUE
C
      DO 105 I=1,IJ
C
      DO 105 J=1,IJ
      GT(5,I,J)=0.
C
      DO 105 K=1,IJ
      GT(5,I,J)=GT(5,I,J)+GT(3,I,K)*GT(4,K,J)
  105 CONTINUE
C
      DO 106 I=1,IJ
C
      DO 106 J=1,IJ
      GT(6,I,J)=0.
C
      DO 106 K=1,IJ
      GT(6,I,J)=GT(6,I,J)+GT(5,I,K)*GT(3,K,J)
  106 CONTINUE
C
```

```
      DO 107 I=1,IJ
C
      DO 107 J=1,IJ
      GT(7,I,J)=0.
C
      DO 107 K=1,IJ
      GT(7,I,J)=GT(7,I,J)+GT(5,I,K)*GT(6,K,J)
  107 CONTINUE
C
      DO 108 I=1,IJ
C
      DO 108 J=1,IJ
      GT(8,I,J)=0.
C
      DO 108 K=1,IJ
      GT(8,I,J)=GT(8,I,J)+GT(5,I,K)*GT(7,K,J)
  108 CONTINUE
C
      DO 109 I=1,IJ
C
      DO 109 J=1,IJ
      GT(9,I,J)=0.
C
      DO 109 K=1,IJ
      GT(9,I,J)=GT(9,I,J)+GT(7,I,K)*GT(8,K,J)
  109 CONTINUE
C
      DO 110 I=1,IJ
C
      DO 110 J=1,IJ
      GT(10,I,J)=0.
C
      DO 110 K=1,IJ
      GT(10,I,J)=GT(10,I,J)+GT(7,I,K)*GT(9,K,J)
  110 CONTINUE
C
      DO 111 I=1,IJ
C
      DO 111 J=1,IJ
      GT(11,I,J)=0.
C
      DO 111 K=1,IJ
      GT(11,I,J)=GT(11,I,J)+GT(7,I,K)*GT(10,K,J)
  111 CONTINUE
C
      DO 112 I=1,IJ
C
      DO 112 J=1,IJ
      GT(12,I,J)=0.
C
      DO 112 K=1,IJ
      GT(12,I,J)=GT(12,I,J)+GT(9,I,K)*GT(11,K,J)
  112 CONTINUE
C
      DO 200 IX=1,IL
C
      DO 200 I=1,IJ
C
      DO 200 J=1,IJ
      IF(I.GT.J) GO TO 200
      IF(I.LT.J) GO TO 201
      GT(IX,I,J)=ALOG(GT(IX,I,J))
      GO TO 200
  201 CONTINUE
      IF(TS(E0,I,J).LE.0.)GO TO 200
      IF(IX.EQ.1) GT(IX,I,J)=ALOG(TS(E0,I,J))
```

```
      IF(IX.GT.1) GT(IX,I,J)=ALOG(GT(IX,I,J)/XT(IX))
200   CONTINUE
C
      GO TO 2
      END
C
C ***********************************************
C
      SUBROUTINE GPERT(GG0,GG1,GG2,X)
C
      PARAMETER(IJ=59)
C
      DIMENSION GG0(IJ,IJ),GG1(IJ,IJ),GG2(IJ,IJ)
C
      E0=2000.
C
      DO 1 I=1,IJ
C
      DO 1 J=1,IJ
      GG0(I,J)=0.
      IF(I.EQ.J) GG0(I,J)=TEXP(-(XS(E0,I)-TS(E0,J,J))*X)
    1 CONTINUE
C
      DO 2 I=1,IJ
C
      DO 2 J=1,IJ
      GG1(I,J)=0.
      IF(I.NE.J) GG1(I,J)=TS(E0,I,J)*BUILD(XS(E0,J)-TS(E0,J,J),
     -XS(E0,I)-TS(E0,I,I),X)
    2 CONTINUE
C
      DO 3 I=1,IJ
C
      DO 3 J=1,IJ
      GG2(I,J)=0.
      IF(I.GE.J) GO TO 3
C
      DO 3 K=I+1,J-1
      GG2(I,J)=GG2(I,J)+TS(E0,I,K)*TS(E0,K,J)*G2(XS(E0,J)-TS(E0,J,J),
     -XS(E0,K)-TS(E0,K,K),XS(E0,I)-TS(E0,I,I),X)
    3 CONTINUE
C
      RETURN
      END
C
C ***********************************************
C
      FUNCTION BUILD(A,B,X)
C
      IF(ABS(A-B).LT.0001*A)GO TO 1
      BUILD=(TEXP(-A*X)-TEXP(-B*X))/(B-A)
      RETURN
    1 CONTINUE
      BUILD=X*TEXP(-(A+B)*X/2.)
      RETURN
      END
C
C ***********************************************
C
      FUNCTION RMAT(E,A,Z)
C
C   FUNCTION RMAT GENERATES THE RANGE, ENERGY, AND LET ARRAYS FOR
C   ARBITRARY IONS IN TARGET. E IS ION ENERGY IN MEV/AMU
C
      EXTERNAL ATOE
C
```

```
      DIMENSION IOR(2),IP1(2),ROFZ(15)
      DIMENSION ET(60),RT(60,15),ST(60,15),ZT(15),ANS(15),F(15)
      DIMENSION AA(5),ZZ(5),DD(5),EN(60)
      DIMENSION IENDSW(8),ENDX1(15),ENDXN(15),ENDY1(60),ENDYN(60),
     -ENDXY(4),VAL(6,1),WK(2790)
C
C   WK SIZE=3*NX*NY+2*NY+NX=3*60*15+2*15+60=2790
C
      COMMON/TARGET/NLAY,XLAY,DN,NAT,ATRG(5),ZTRG(5),DENSTRG(5)
      COMMON/CONST/ENDX1
C
      DATA IW/0/
      DATA IENDSW/0,1,1,1,1,1,1,1/
      DATA IOR/2,2/
      DATA IP1,IP3/-1,-1,-1/
      DATA EN/.01,.02,.03,.04,.05,.06,.07,.08,.09,.1,.2,.3,.4,.5,
     -.6,.7,.8,.9,1.,2.,3.,4.,5.,6.,7.,8.,9.,10.,20.,30.,40.,50.,
     -60.,70.,80.,90.,100.,150.,200.,300.,400.,500.,600.,700.,
     -800.,900.,1000.,1500.,2000.,2500.,3000.,4000.,5000.,6000.,
     -7000.,8500.,10000.,20000.,35000.,50000./
      DATA ZT/1.,2.,3.5,5.,7.,10.,14.,22.,30.,40.,50.,
     -60.,70.,80.,92./
      DATA NOLD,NP/0,60/
C
C   RANGE INTERPOLATION/EXTRAPOLATION OF RT(ET,ZT)
C
      N=1
      IF(NOLD.NE.NLAY) GO TO 150
   10 CONTINUE
      RMAT=0.
      IF(E.LE.0.) RETURN
      IF(E.GT..01)GO TO 20
      EL=ALOG(0.01)
      CALL IBI(60,ET,15,ZT,60,RT,IOR,IP1,EL,Z,RMAT,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RMAT FAILED WITH ERROR=',IER
      STOP
      END IF
      CALL IUNI(15,15,ZT,1,ENDX1,IOR,Z,RVAL,IP3,IER)
      IF(IER.NE.0.AND.IER.NE.-4) THEN
      PRINT *,' IUNI IN MODULE RMAT FAILED WITH ERROR=',IER
      STOP
      END IF
      RMAT=A*TEXP(RMAT)/(Z*Z)
      RMAT=RMAT*(E/.01)**(1.-RVAL)
      RETURN
   20 CONTINUE
      IF(E.LT.50000.) GO TO 30
      ELP=ALOG(35000.)
      EL=ALOG(50000.)
      CALL IBI(60,ET,15,ZT,60,RT,IOR,IP1,ELP,Z,RMATP,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RMAT FAILED WITH ERROR=',IER
      STOP
      END IF
      CALL IBI(60,ET,15,ZT,60,RT,IOR,IP1,EL,Z,RMAT,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RMAT FAILED WITH ERROR=',IER
      STOP
      END IF
      RMATP=A*TEXP(RMATP)/(Z*Z)
      RMAT=A*TEXP(RMAT)/(Z*Z)
      RVAL=ALOG((50000.*RMATP)/(35000.*RMAT))/ALOG(50000./35000.)
      RMAT=RMAT*((E/50000.)**(1.-RVAL))
      RETURN
   30 CONTINUE
```

```
      EL=ALOG(E)
      CALL IBI(60,ET,15,ZT,60,RT,IOR,IP1,EL,Z,RMAT,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RMAT FAILED WITH ERROR=',IER
      STOP
      END IF
      RMAT=A*TEXP(RMAT)/(Z*Z)
      RETURN
C
C   STOPPING POWER INTERPOLATION/EXTRAPOLATION OF ST(ET,ZT)
C
      ENTRY SMAT
      N=2
      IF(NOLD.NE.NLAY) GO TO 150
   40 CONTINUE
      RMAT=0.
      IF(E.LE.0.) RETURN
      IF(E.GT..01)GO TO 50
      EL=ALOG(0.01)
      CALL IBI(60,ET,15,ZT,60,ST,IOR,IP1,EL,Z,RMAT,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RMAT FAILED WITH ERROR=',IER
      STOP
      END IF
      CALL IUNI(15,15,ZT,1,ENDX1,IOR,Z,RVAL,IP3,IER)
      IF(IER.NE.0.AND.IER.NE.-4) THEN
      PRINT *,' IUNI IN MODULE RMAT FAILED WITH ERROR=',IER
      STOP
      END IF
      RMAT=Z*Z*TEXP(RMAT)
      RMAT=RMAT*(E/.01)**RVAL
      RETURN
   50 CONTINUE
      IF(E.LT.50000.) GO TO 60
      ELP=ALOG(35000.)
      EL=ALOG(50000.)
      CALL IBI(60,ET,15,ZT,60,ST,IOR,IP1,ELP,Z,RMATP,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RMAT FAILED WITH ERROR=',IER
      STOP
      END IF
      CALL IBI(60,ET,15,ZT,60,ST,IOR,IP1,EL,Z,RMAT,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RMAT FAILED WITH ERROR=',IER
      STOP
      END IF
      RMATP=Z*Z*TEXP(RMATP)
      RMAT=Z*Z*TEXP(RMAT)
      RMAT=RMAT+(RMATP-RMAT)*(ALOG(E)-EL)/(ELP-EL)
      RETURN
   60 CONTINUE
      EL=ALOG(E)
      CALL IBI(60,ET,15,ZT,60,ST,IOR,IP1,EL,Z,RMAT,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RMAT FAILED WITH ERROR=',IER
      STOP
      END IF
      RMAT=Z*Z*TEXP(RMAT)
      RETURN
C
C   D(STOPPING POWER)/D(E) CALCULATION, INTERPOLATION/EXTRAPOLATION
C   OF ST(ET,ZT)
C
      ENTRY DSMAT
      N=3
      IF(NOLD.NE.NLAY) GO TO 150
```

```
 70 CONTINUE
    RMAT=0.
    IF(E.LE.0.) RETURN
    IF(E.GT..01) GO TO 80
    EL=ALOG(.01)
    CALL SUTPAR(60,15,ET,ZT,60,ST,IENDSW,ENDX1,ENDXN,ENDY1,ENDYN,
   _            ENDXY,1.,1,EL,Z,IW,VAL,WK,IER)
    IF(IER.NE.0) THEN
    PRINT *,' SUTPAR IN MODULE RMAT FAILED WITH ERROR=',IER
    STOP
    END IF
    CALL IUNI(15,15,ZT,1,ENDX1,IOR,Z,RVAL,IP3,IER)
    IF(IER.NE.0.AND.IER.NE.-4) THEN
    PRINT *,' IUNI IN MODULE RMAT FAILED WITH ERROR=',IER
    STOP
    END IF
    RMAT1=VAL(1,1)
    RMAT1=Z*Z*TEXP(RMAT1)
    RMAT2=VAL(2,1)
    RMAT=RMAT1*RMAT2/.01
    RMAT=RMAT*(E/.01)**(RVAL-1.)
    RETURN
 80 CONTINUE
    IF(E.LT.50000.) GO TO 90
    EL=ALOG(50000.)
    CALL SUTPAR(60,15,ET,ZT,60,ST,IENDSW,ENDX1,ENDXN,ENDY1,ENDYN,
   _            ENDXY,1.,1,EL,Z,IW,VAL,WK,IER)
    IF(IER.NE.0) THEN
    PRINT *,' SUTPAR IN MODULE RMAT FAILED WITH ERROR=',IER
    STOP
    END IF
    RMAT1=VAL(1,1)
    RMAT1=Z*Z*TEXP(RMAT1)
    RMAT2=VAL(2,1)
    RMAT=RMAT1*RMAT2/50000.
    RMAT=RMAT*50000./E
    RETURN
 90 CONTINUE
    EL=ALOG(E)
    CALL SUTPAR(60,15,ET,ZT,60,ST,IENDSW,ENDX1,ENDXN,ENDY1,ENDYN,
   _            ENDXY,1.,1,EL,Z,IW,VAL,WK,IER)
    IF(IER.NE.0) THEN
    PRINT *,' SUTPAR IN MODULE RMAT FAILED WITH ERROR=',IER
    STOP
    END IF
    RMAT1=VAL(1,1)
    RMAT1=Z*Z*TEXP(RMAT1)
    RMAT2=VAL(2,1)
    RMAT=RMAT1*RMAT2/E
    RETURN
C
C   ENERGY INTERPOLATION/EXTRAPOLATION ET(ZT,RT)
C
    ENTRY EMAT
    N=4
    IF(NOLD.NE.NLAY) GO TO 150
100 CONTINUE
    RMAT=0.
    RL=E
    IF(RL.LE.0.) RETURN
    R=ALOG(Z*Z*RL/A)
C
    DO 130 I=1,15
    CALL IUNI(60,60,RT(1,I),1,ET,IOR,R,ROFZ(I),IP3,IER)
    IF(IER.NE.0.AND.IER.NE.-4) THEN
    PRINT *,' IUNI IN MODULE RMAT FAILED WITH ERROR=',IER
```

```
      STOP
      END IF
  130 CONTINUE
C
      CALL IUNI(15,15,ZT,1,ROFZ,IOR,Z,RMAT,IP3,IER)
      IF(IER.NE.0.AND.IER.NE.-4) THEN
      PRINT *,' IUNI IN MODULE RMAT FAILED WITH ERROR=',IER
      STOP
      END IF
      RMAT=TEXP(RMAT)
      IF(RMAT.GE..01.AND.RMAT.LE.50000.) RETURN
      IF(RMAT.GT.50000.) GO TO 140
C
      DO 135 I=1,15
      ROFZ(I)=RT(1,1)
  135 CONTINUE
C
      CALL IUNI(15,15,ZT,1,ROFZ,IOR,Z,RMAT,IP3,IER)
      IF(IER.NE.0.AND.IER.NE.-4) THEN
      PRINT *,' IUNI IN MODULE RMAT FAILED WITH ERROR=',IER
      STOP
      END IF
      CALL IUNI(15,15,ZT,1,ENDX1,IOR,Z,RVAL,IP3,IER)
      IF(IER.NE.0.AND.IER.NE.-4) THEN
      PRINT *,' IUNI IN MODULE RMAT FAILED WITH ERROR=',IER
      STOP
      END IF
      RMAT=A*TEXP(RMAT)/(Z*Z)
      RMAT=0.01*(RL/RMAT)**(1./(1.-RVAL))
      RETURN
  140 CONTINUE
C
      ELP=ALOG(35000.)
      EL=ALOG(50000.)
      CALL IBI(60,ET,15,ZT,60,RT,IOR,IP1,ELP,Z,RMATP,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE EMAT FAILED WITH ERROR=',IER
      STOP
      END IF
      CALL IBI(60,ET,15,ZT,60,RT,IOR,IP1,EL,Z,RMAT,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE EMAT FAILED WITH ERROR=',IER
      STOP
      END IF
      RMATP=A*TEXP(RMATP)/(Z*Z)
      RMAT=A*TEXP(RMAT)/(Z*Z)
      RVAL=ALOG((50000.*RMATP)/(35000.*RMAT))/ALOG(50000./35000.)
      RMAT=50000.*(RL/RMAT)**(1./(1.-RVAL))
      RETURN
C
  150 CONTINUE
      OPEN(101,FILE='ATOMIC.DAT',STATUS='OLD')
      REWIND (101)
  190 READ (101,160,END=170)DDNN,NN,AA,ZZ,DD,RT,ST,ET
      PRINT 160,DDNN,NN,AA,ZZ,DD
      PRINT *,' RMAT FOUND ',NN,' ELEMENTS OF CHARGE ',(ZZ(I),I=1,NN)
  160 FORMAT(F10.3,I5/5F10.3/5F10.1/5E12.3/180(5E13.4/)/180(5E13.4/)/
     -18(5E13.4/))
      IF(DDNN.NE.DN)GO TO 190
      IF(NN.NE.NAT) GO TO 190
C
      DO 180 I=1,NN
      IF(I.GT.NAT) GO TO 190
      IF(AA(I).NE.ATRG(I)) GO TO 190
      IF(ZZ(I).NE.ZTRG(I)) GO TO 190
      IF(DD(I).NE.DENSTRG(I)) GO TO 190
```

```
  180 CONTINUE
C
      PRINT *,' MATERIAL FOR RMAT ',NLAY,' FOUND ON ATOMIC FILE '
     1,' THERE ARE ',NAT,' ELEMENTS OF CHARGES ',(ZTRG(I),I=1,NAT)
      GO TO 200
  170 CONTINUE
      PRINT *,' ALL ATOMIC DATA FAILED RMAT TEST  '
      CALL MGAUSS(1.E-6,EN(1),6,ANS,ATOE,F,15)
C
      DO 210 I=1,15
      V=ZT(I)
      W=2.*V
      RT(1,I)=ANS(I)
C
      DO 210 J=1,NP
      ST(J,I)=SIONA(EN(J),W,V)
  210 CONTINUE
C
      DO 220 I=2,NP
      CALL MGAUSS(EN(I-1),EN(I),6,ANS,ATOE,F,15)
C
      DO 220 J=1,15
      RT(I,J)=RT(I-1,J)+ANS(J)
  220 CONTINUE
C
      DO 240 I=1,NP
      ET(I)=ALOG(EN(I))
C
      DO 240 J=1,15
      V=ZT(J)
      W=2.*V
      RT(I,J)=ALOG(V*V*RT(I,J)/W)
      ST(I,J)=ALOG(ST(I,J)/(V*V))
  240 CONTINUE
C
      OPEN(103,FILE='NEWRMAT.DAT',STATUS='NEW')
      WRITE(103,160)DN,NAT,ATRG,ZTRG,DENSTRG,RT,ST,ET
      PRINT *,' MATERIAL FOR RMAT ',NLAY,' PLACED ON NEWRMAT FILE'
     1,' THERE ARE ',NAT,' ELEMENTS OF CHARGES ',(ZTRG(I),I=1,NAT)
      CLOSE(103,STATUS='KEEP')
  200 CONTINUE
      CLOSE(101,STATUS='KEEP')
C
      NOLD=NLAY
C
      DO 250 I=1,15
      DS1=(ST(2,I)-ST(1,I))/(ET(2)-ET(1))
      DS2=(ST(3,I)-ST(2,I))/(ET(3)-ET(2))
      ENDX1(I)=DS1+(DS2-DS1)*(ET(1)-ET(2))/(ET(3)-ET(1))
  250 CONTINUE
C
      PRINT 260
  260 FORMAT('  RMAT IS INITIALIZED')
      GO TO (10,40,70,100),N
      END
C
C************************FUNCTION RTIS(E,A,Z)************************
C
      FUNCTION RTIS(E,A,Z)
C
C   FUNCTION RTIS GENERATES THE RANGE, ENERGY, AND LET ARRAYS FOR
C   ARBITRARY IONS IN TARGET. E IS ION ENERGY IN MEV/AMU
C
      EXTERNAL ATOE
C
      DIMENSION IOR(2),IP1(2),ROFZ(15)
```

```
      DIMENSION ET(60),RT(60,15),ST(60,15),ZT(15),ANS(15),F(15)
      DIMENSION AA(5),ZZ(5),DD(5),EN(60)
      DIMENSION IENDSW(8),ENDX1(15),ENDXN(15),ENDY1(60),ENDYN(60),
     -ENDXY(4),VAL(6,1),WK(2790)
C
C     WK SIZE=3*NX*NY+2*NY+NX=3*60*15+2*15+60=2790
C
      COMMON/TARGET/NLAY,XLAY,DN,NAT,ATRG(5),ZTRG(5),DENSTRG(5)
      COMMON/CONST/ENDX1
C
      DATA IW/0/
      DATA IENDSW/0,1,1,1,1,1,1,1/
      DATA IOR/2,2/
      DATA IP1,IP3/-1,-1,-1/
      DATA EN/.01,.02,.03,.04,.05,.06,.07,.08,.09,.1,.2,.3,.4,.5,
     -.6,.7,.8,.9,1.,2.,3.,4.,5.,6.,7.,8.,9.,10.,20.,30.,40.,50.,
     -60.,70.,80.,90.,100.,150.,200.,300.,400.,500.,600.,700.,
     -800.,900.,1000.,1500.,2000.,2500.,3000.,4000.,5000.,6000.,
     -7000.,8500.,10000.,20000.,35000.,50000./
      DATA ZT/1.,2.,3.5,5.,7.,10.,14.,22.,30.,40.,50.,
     -60.,70.,80.,92./
      DATA NOLD,NP/0,60/
C
C     RANGE INTERPOLATION/EXTRAPOLATION OF RT(ET,ZT)
C
C
      N=1
      IF(NOLD.EQ.0) GO TO 150
   10 CONTINUE
      RTIS=0.
      IF(E.LE.0.) RETURN
      IF(E.GT..01)GO TO 20
      EL=ALOG(0.01)
      CALL IBI(60,ET,15,ZT,60,RT,IOR,IP1,EL,Z,RTIS,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      CALL IUNI(15,15,ZT,1,ENDX1,IOR,Z,RVAL,IP3,IER)
      IF(IER.NE.0.AND.IER.NE.-4) THEN
      PRINT *,' IUNI IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      RTIS=A*TEXP(RTIS)/(Z*Z)
      RTIS=RTIS*(E/.01)**(1.-RVAL)
      RETURN
   20 CONTINUE
      IF(E.LT.50000.) GO TO 30
      ELP=ALOG(35000.)
      EL=ALOG(50000.)
      CALL IBI(60,ET,15,ZT,60,RT,IOR,IP1,ELP,Z,RTISP,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      CALL IBI(60,ET,15,ZT,60,RT,IOR,IP1,EL,Z,RTIS,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      RTISP=A*TEXP(RTISP)/(Z*Z)
      RTIS=A*TEXP(RTIS)/(Z*Z)
      RVAL=ALOG((50000.*RTISP)/(35000.*RTIS))/ALOG(50000./35000.)
      RTIS=RTIS*((E/50000.)**(1.-RVAL))
      RETURN
   30 CONTINUE
      EL=ALOG(E)
```

```
      CALL IBI(60,ET,15,ZT,60,RT,IOR,IP1,EL,Z,RTIS,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      RTIS=A*TEXP(RTIS)/(Z*Z)
      RETURN
C
C     STOPPING POWER INTERPOLATION/EXTRAPOLATION OF ST(ET,ZT)
C
      ENTRY STIS
      N=2
      IF(NOLD.EQ.0) GO TO 150
   40 CONTINUE
      RTIS=0.
      IF(E.LE.0.) RETURN
      IF(E.GT..01)GO TO 50
      EL=ALOG(0.01)
      CALL IBI(60,ET,15,ZT,60,ST,IOR,IP1,EL,Z,RTIS,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      CALL IUNI(15,15,ZT,1,ENDX1,IOR,Z,RVAL,IP3,IER)
      IF(IER.NE.0.AND.IER.NE.-4) THEN
      PRINT *,' IUNI IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      RTIS=Z*Z*TEXP(RTIS)
      RTIS=RTIS*(E/.01)**RVAL
      RETURN
   50 CONTINUE
      IF(E.LT.50000.) GO TO 60
      ELP=ALOG(35000.)
      EL=ALOG(50000.)
      CALL IBI(60,ET,15,ZT,60,ST,IOR,IP1,ELP,Z,RTISP,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      CALL IBI(60,ET,15,ZT,60,ST,IOR,IP1,EL,Z,RTIS,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      RTISP=Z*Z*TEXP(RTISP)
      RTIS=Z*Z*TEXP(RTIS)
      RTIS=RTIS+(RTISP-RTIS)*(ALOG(E)-EL)/(ELP-EL)
      RETURN
   60 CONTINUE
      EL=ALOG(E)
      CALL IBI(60,ET,15,ZT,60,ST,IOR,IP1,EL,Z,RTIS,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      RTIS=Z*Z*TEXP(RTIS)
      RETURN
C
C     D(STOPPING POWER)/D(E) CALCULATION, INTERPOLATION/EXTRAPOLATION
C     OF ST(ET,ZT)
C
      ENTRY DSTIS
      N=3
      IF(NOLD.EQ.0) GO TO 150
   70 CONTINUE
```

```
      RTIS=0.
      IF(E.LE.0.) RETURN
      IF(E.GT..01) GO TO 80
      EL=ALOG(.01)
      CALL SUTPAR(60,15,ET,ZT,60,ST,IENDSW,ENDX1,ENDXN,ENDY1,ENDYN,
     _            ENDXY,1.,1,EL,Z,IW,VAL,WK,IER)
      IF(IER.NE.0) THEN
      PRINT *,' SUTPAR IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      CALL IUNI(15,15,ZT,1,ENDX1,IOR,Z,RVAL,IP3,IER)
      IF(IER.NE.0.AND.IER.NE.-4) THEN
      PRINT *,' IUNI IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      RTIS1=VAL(1,1)
      RTIS1=Z*Z*TEXP(RTIS1)
      RTIS2=VAL(2,1)
      RTIS=RTIS1*RTIS2/.01
      RTIS=RTIS*(E/.01)**(RVAL-1.)
      RETURN
   80 CONTINUE
      IF(E.LT.50000.) GO TO 90
      EL=ALOG(50000.)
      CALL SUTPAR(60,15,ET,ZT,60,ST,IENDSW,ENDX1,ENDXN,ENDY1,ENDYN,
     _            ENDXY,1.,1,EL,Z,IW,VAL,WK,IER)
      IF(IER.NE.0) THEN
      PRINT *,' SUTPAR IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      RTIS1=VAL(1,1)
      RTIS1=Z*Z*TEXP(RTIS1)
      RTIS2=VAL(2,1)
      RTIS=RTIS1*RTIS2/50000.
      RTIS=RTIS*50000./E
      RETURN
   90 CONTINUE
      EL=ALOG(E)
      CALL SUTPAR(60,15,ET,ZT,60,ST,IENDSW,ENDX1,ENDXN,ENDY1,ENDYN,
     _            ENDXY,1.,1,EL,Z,IW,VAL,WK,IER)
      IF(IER.NE.0) THEN
      PRINT *,' SUTPAR IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      RTIS1=VAL(1,1)
      RTIS1=Z*Z*TEXP(RTIS1)
      RTIS2=VAL(2,1)
      RTIS=RTIS1*RTIS2/E
      RETURN
C
C   ENERGY INTERPOLATION/EXTRAPOLATION ET(ZT,RT)
C
      ENTRY ETIS
      N=4
      IF(NOLD.EQ.0) GO TO 150
  100 CONTINUE
      RTIS=0.
      RL=E
      IF(RL.LE.0.) RETURN
      R=ALOG(Z*Z*RL/A)
C
      DO 130 I=1,15
      CALL IUNI(60,60,RT(1,I),1,ET,IOR,R,ROFZ(I),IP3,IER)
      IF(IER.NE.0.AND.IER.NE.-4) THEN
      PRINT *,' IUNI IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
```

```
      END IF
  130 CONTINUE
C
      CALL IUNI(15,15,ZT,1,ROFZ,IOR,Z,RTIS,IP3,IER)
      IF(IER.NE.0.AND.IER.NE.-4) THEN
      PRINT *,' IUNI IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      RTIS=TEXP(RTIS)
      IF(RTIS.GE..01.AND.RTIS.LE.50000.) RETURN
      IF(RTIS.GT.50000.) GO TO 140
C
      DO 135 I=1,15
      ROFZ(I)=RT(1,I)
  135 CONTINUE
C
      CALL IUNI(15,15,ZT,1,ROFZ,IOR,Z,RTIS,IP3,IER)
      IF(IER.NE.0.AND.IER.NE.-4) THEN
      PRINT *,' IUNI IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      CALL IUNI(15,15,ZT,1,ENDX1,IOR,Z,RVAL,IP3,IER)
      IF(IER.NE.0.AND.IER.NE.-4) THEN
      PRINT *,' IUNI IN MODULE RTIS FAILED WITH ERROR=',IER
      STOP
      END IF
      RTIS=A*TEXP(RTIS)/(Z*Z)
      RTIS=0.01*(RL/RTIS)**(1./(1.-RVAL))
      RETURN
  140 CONTINUE
C
      ELP=ALOG(35000.)
      EL=ALOG(50000.)
      CALL IBI(60,ET,15,ZT,60,RT,IOR,IP1,ELP,Z,RTISP,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE ETIS FAILED WITH ERROR=',IER
      STOP
      END IF
      CALL IBI(60,ET,15,ZT,60,RT,IOR,IP1,EL,Z,RTIS,IER)
      IF(IER.NE.0.AND.IER.NE.-3) THEN
      PRINT *,' IBI IN MODULE ETIS FAILED WITH ERROR=',IER
      STOP
      END IF
      RTISP=A*TEXP(RTISP)/(Z*Z)
      RTIS=A*TEXP(RTIS)/(Z*Z)
      RVAL=ALOG((50000.*RTISP)/(35000.*RTIS))/ALOG(50000./35000.)
      RTIS=50000.*(RL/RTIS)**(1./(1.-RVAL))
      RETURN
C
  150 CONTINUE
      OPEN(101,FILE='ATOMIC.DAT',STATUS='OLD')
      REWIND (101)
  190 READ (101,160,END=170)DDNN,NN,AA,ZZ,DD,RT,ST,ET
      PRINT 160,DDNN,NN,AA,ZZ,DD
      PRINT *,' RTIS FOUND ',NN,' ELEMENTS OF CHARGE ',(ZZ(I),I=1,NN)
  160 FORMAT(F10.3,I5/5F10.3/5F10.1/5E12.3/180(5E13.4/)/180(5E13.4/)/
     -18(5E13.4/))
      IF(DDNN.NE.DN)GO TO 190
      IF(NN.NE.NAT) GO TO 190
C
      DO 180 I=1,NN
      IF(I.GT.NAT) GO TO 190
      IF(AA(I).NE.ATRG(I)) GO TO 190
      IF(ZZ(I).NE.ZTRG(I)) GO TO 190
      IF(DD(I).NE.DENSTRG(I)) GO TO 190
  180 CONTINUE
```

```
C
      PRINT *,' MATERIAL FOR RTIS ',NLAY,' FOUND ON ATOMIC FILE '
     1,' THERE ARE ',NAT,' ELEMENTS OF CHARGES ',(ZTRG(I),I=1,NAT)
      GO TO 200
  170 CONTINUE
      PRINT *,' ALL ATOMIC DATA FAILED RTIS TEST  '
      CALL MGAUSS(1.E-6,EN(1),6,ANS,ATOE,F,15)
C
      DO 210 I=1,15
      V=ZT(I)
      W=2.*V
      RT(1,I)=ANS(I)
C
      DO 210 J=1,NP
      ST(J,I)=SIONA(EN(J),W,V)
  210 CONTINUE
C
      DO 220 I=2,NP
      CALL MGAUSS(EN(I-1),EN(I),6,ANS,ATOE,F,15)
C
      DO 220 J=1,15
      RT(I,J)=RT(I-1,J)+ANS(J)
  220 CONTINUE
C
      DO 240 I=1,NP
      ET(I)=ALOG(EN(I))
C
      DO 240 J=1,15
      V=ZT(J)
      W=2.*V
      RT(I,J)=ALOG(V*V*RT(I,J)/W)
      ST(I,J)=ALOG(ST(I,J)/(V*V))
  240 CONTINUE
C
      OPEN(103,FILE='NEWRTIS.DAT',STATUS='NEW')
      WRITE(103,160)DN,NAT,ATRG,ZTRG,DENSTRG,RT,ST,ET
      PRINT *,' MATERIAL FOR RTIS ',NLAY,' PLACED ON NEWRTIS FILE'
     1,' THERE ARE ',NAT,' ELEMENTS OF CHARGES ',(ZTRG(I),I=1,NAT)
      CLOSE(103,STATUS='KEEP')
  200 CONTINUE
      CLOSE(101,STATUS='KEEP')
C
      NOLD=NLAY
C
      DO 250 I=1,15
      DS1=(ST(2,I)-ST(1,I))/(ET(2)-ET(1))
      DS2=(ST(3,I)-ST(2,I))/(ET(3)-ET(2))
      ENDX1(I)=DS1+(DS2-DS1)*(ET(1)-ET(2))/(ET(3)-ET(1))
  250 CONTINUE
C
      PRINT 260
  260 FORMAT('  RTIS IS INITIALIZED')
      GO TO (10,40,70,100),N
      END
C
C*********************SUBROUTINE ATOE(E,F)*****************************
C
      SUBROUTINE ATOE (E,F)
C
C   SUBROUTINE ATOE IS THE INTEGRAND OF RANGE FORMULA
C
      DIMENSION ZT(15),F(15)
C
      DATA ZT/1.,2.,3.5,5.,7.,10.,14.,22.,30.,40.,50.,
     -60.,70.,80.,92./
C
```

```
      DO 10 I=1,15
      V=ZT(I)
      W=2.*V
      F(I)=W/SIONA(E,W,V)
   10 CONTINUE
C
      RETURN
      END
C
C*********************FUNCTION SIONA(E,A,Z)*****************************
C
      FUNCTION SIONA(E,A,Z)
C
C   E IS IN UNITS OF MEV/AMU
C
      IF(Z.EQ.1.) THEN
      CALL ATOPP(E,S)
      SIONA=S
      CALL ATOPN(E,SN,A,Z)
      SIONA=SIONA+SN
      RETURN
      ELSE
      CALL ATOPA(E,S,Z)
      SIONA=S*(ZEFF(E,A,Z)/ZEFF(E,4.,2.))**2
      CALL ATOPN(E,SN,A,Z)
      SIONA=SIONA+SN
      RETURN
      END IF
      END
C
C*********************FUNCTION ZEFF(E,A,Z)*****************************
C
      FUNCTION ZEFF(E,A,Z)
C
C   FUNCTION ZEFF CALCULATES ION EFFECTIVE CHARGE
C
      IF(E.LT.1.E-10) E=1.E-10
      ARG=1./((1.+E/938.3)**2)
      IF(ARG.EQ.1.) THEN
      ZEFF=Z*(1-TEXP(-5.77*SQRT(E)/Z**(2./3.)))
      ELSE
      BETA=SQRT(1.-ARG)
      ZEFF=Z*(1-TEXP(-125.*BETA/Z**(2./3.)))
      END IF
      RETURN
      END
C
C*********************SUBROUTINE ATOPP(T,S)*****************************
C
      SUBROUTINE ATOPP(T,S)
C
C   SUBROUTINE ATOPP CALCULATES PROTON STOPPING POWER
C
      DIMENSION A(12,5),STP(5)
C
      COMMON/TARGET/NLAY,XLAY,DN,NAT,ATRG(5),ZTRG(5),DENSTRG(5)
C
      DATA NOLD/0/
C
      IF(NOLD.NE.NLAY) GO TO 100
C
C   CONVERT T IN MEV TO E IN KEV
C
   10 E=1000.*T
C
C   ENERGY 0<E<=10 KEV
```

```
C
      IF(E.LE.10.) THEN
C
      DO 50 I=1,NAT
      STP(I)=A(1,I)*SQRT(E)
   50 CONTINUE
C
C    ENERGY 10<E<1000 KEV
C
      ELSE IF(E.LT.1000.) THEN
C
      DO 70 I=1,NAT
      SL=A(2,I)*E**.45
      SH=(A(3,I)/E)*ALOG(1.+A(4,I)/E+A(5,I)*E)
      STP(I)=1./(1./SL+1./SH)
   70 CONTINUE
C
C    ENERGY 1000<=E<=100000 KEV
C
      ELSE
C
C    B2=(V/C)**2 IN ENERGY TERM
C
      B2=1.-1./(1.+E/938300.)**2
C
C    CALCULATE ASYMPTOTIC DENSITY EFFECT
C
      DEL=DENSEFF(T)
C
      DO 30 I=1,NAT
      SUM=0.
C
      DO 20 J=1,5
      L=J-1
      K=J+7
      SUM=SUM+A(K,I)*(ALOG(E)**L)
   20 CONTINUE
C
      SUM=SUM*(1.-TEXP(-2.*(1.E5/E)**2))
      STP(I)=(A(6,I)/B2)*(ALOG(A(7,I)*B2/(1-B2))-B2-SUM-DEL/2.)
   30 CONTINUE
C
      END IF
      S=0.
C
      DO 90 I=1,NAT
      S=S+STP(I)*DENSTRG(I)
   90 CONTINUE
C
      S=S*1.E-21
      RETURN
C
  100 CONTINUE
C
      DO 110 I=1,12
      DO 110 J=1,NAT
      A(I,J)=ACOEF(I,ZTRG(J))
  110 CONTINUE
C
      NOLD=NLAY
      GO TO 10
      END
C
C*****************FUNCTION DENSEFF(E)*****************
C
      FUNCTION DENSEFF(E)
```

```
C
C    SUBROUTINE DENSEFF CALCULATES THE 'ASYMPTOTIC' DENSITY EFFECT
C    OF THE STOPPING POWER (CORRECTION TO BETHE-BLOCH FORMULA)
C
C    E IS THE KINETIC ENERGY OF PROJECTILE IN MEV/AMU
C
       COMMON/TARGET/NLAY,XLAY,DN,NAT,ATRG(5),ZTRG(5),DENSTRG(5)
C
       DATA NOLD/0/
C
C    RYDBERG CONSTANT RY AND ELECTRON REST MASS EMASS ARE IN EV
C
       DATA RY,EMASS,PMASS,A0,PI/13.6,.511E6,938.3,5.2928E-9,3.14159265/
C
       IF(NOLD.NE.NLAY) GO TO 5
     3 CONTINUE
       WP=4.*RY*SQRT(PI*RHOE*(A0**3))
       C=-2.*ALOG(EMEAN/WP)-1.
C
C    ATA=P/MC (KINETIC MOMENTUM/REST MOMENTUM)
C
       ATA=SQRT(E*E+2.*PMASS*E)/PMASS
       DEL=ALOG(ATA**2)+C
       IF(DEL.LT.0.) DEL=0.
       DENSEFF=DEL
       RETURN
     5 CONTINUE
C
C    CALCULATE IONIZATION POTENTIAL DUE TO IONIZATION OF ALL
C    TARGET MATERIALS
C    REMEAN=ALOG OF ALL IONIZATION POTENTIALS
C    RHOE=ELECTRON DENSITY OF ALL IONS
C
       REMEAN=0.
       RHOE=0.
C
       DO 2 J=1,NAT
       RHOE=RHOE+ZTRG(J)*DENSTRG(J)
       REMEAN=REMEAN+ZTRG(J)*DENSTRG(J)*ALOG((2.*EMASS)/ACOEF(7,ZTRG(J)))
     2 CONTINUE
C
       EMEAN=TEXP(REMEAN/RHOE)
       RHOE=DN*RHOE
       NOLD=NLAY
       GO TO 3
       END
C
C**********************SUBROUTINE ATOPA(T,S)**********************
C
       SUBROUTINE ATOPA(T,S)
C
C    SUBROUTINE ATOPA CALCULATES ALPHA STOPPING POWER
C
       DIMENSION A(9,5),B(9,2),C(9,5),STP(5)
C
       COMMON/TARGET/NLAY,XLAY,DN,NAT,ATRG(5),ZTRG(5),DENSTRG(5)
C
       DATA NOLD/0/
C
C    ARRAY B IS BASED ON LOW ENERGY H,O SOLID
C
       DATA B/.9961,.4126,6.92,8.831,2.582,2.371,.5462,-.07932,-.006853,
      -1.766,.5261,37.11,15.24,2.804,3.782,.3734,-.1011,-.007874/
C
       IF(NOLD.NE.NLAY)GO TO 110
    10 CONTINUE
```

```
C
      DO 30 I=1,9
C
      DO 20 J=1,NAT
      A(I,J)=C(I,J)
      IF(ZTRG(J).EQ.1.) A(I,J)=B(I,1)
      IF(ZTRG(J).EQ.8.) A(I,J)=B(I,2)
   20 CONTINUE
C
   30 CONTINUE
C
C    CONVERT T IN MEV TO E IN KEV
C
      E=1000.*T*4.
C
C    ENERGY E<1 KEV
C
      IF(E.LT.1.) THEN
C
      DO 60 I=1,NAT
      STP(I)=A(1,I)*E**A(2,I)
   60 CONTINUE
C
C    ENERGY 1<=E<=10000 KEV
C
      ELSE IF(E.LE.10000.) THEN
C
      DO 80 I=1,NAT
      SLOW=A(1,I)*E**A(2,I)
      SHIG=A(3,I)/(E/1000.)*ALOG(1.+A(4,I)/(E/1000.)+A(5,I)*E/1000.)
      STP(I)=SLOW*SHIG/(SLOW+SHIG)
   80 CONTINUE
C
C    ENERGY E>=10000 KEV
C
      ELSE
      FE=ALOG(1000./E)
C
      DO 40 I=1,NAT
      STP(I)=TEXP(A(6,I)+A(7,I)*FE+A(8,I)*FE**2+A(9,I)*FE**3)
   40 CONTINUE
C
      END IF
      S=0.
C
      DO 100 I=1,NAT
      S=S+STP(I)*DENSTRG(I)
  100 CONTINUE
C
      S=S*1.E-21
      ACO=1.-(E-2.E4)/2.E4
      IF(E.LT.2.E4) RETURN
      IF(E.GT.4.E4) ACO=0.
      BCO=1.-ACO
      CALL ATOPP(T,STPB)
      S=ACO*S+4.*BCO*STPB
      RETURN
  110 CONTINUE
C
      DO 120 I=1,9
C
      DO 120 J=1,NAT
      C(I,J)=CCOEF(I,ZTRG(J))
  120 CONTINUE
C
      NOLD=NLAY
```

```
      GO TO 10
      END
C
C**********************SUBROUTINE ATOPN(T,S,A1,Z1)***************
C
      SUBROUTINE ATOPN(T,S,A1,Z1)
C
C   SUBROUTINE ATOPN CALCULATES NUCLEAR STOPPING POWER OF
C   ARBITARY IONS
C
      DIMENSION SN(5),A(5),Z(5),STP(5)
C
      COMMON/TARGET/NLAY,XLAY,DN,NAT,ATRG(5),ZTRG(5),DENSTRG(5)
C
C   CONVERT T IN MEV TO E IN KEV
C
      E=1000.*T*A1
C
      DO 40 I=1,NAT
      A(I)=ATRG(I)
      Z(I)=ZTRG(I)
C
C   EPRM IS REDUCED ION ENERGY
C
      EPRM=32.53*A(I)*E/(Z1*Z(I)*(A1+A(I))*(Z1**(2./3.)+Z(I)**(2./3.))
     -**.5)
C
C   REDUCED ION ENERGY EPRM <.01
C
      IF(EPRM.LT.0.01) THEN
      SN(I)=1.593*EPRM**.5
C
C   REDUCED ION ENERGY .01<=EPRM<=10
C
      ELSE IF(EPRM.LE.10.) THEN
      SN(I)=1.7*(EPRM**.5)*(ALOG(EPRM+TEXP(1.))/(1.+6.8*EPRM+3.4*EPRM
     -**1.5))
C
C   REDUCED ION ENERGY EPRM>10
C
      ELSE
      SN(I)=ALOG(.47*EPRM)/(2.*EPRM)
      END IF
   40 CONTINUE
C
C   CONVERT FROM LSS REDUCED STOPPING UNIT TO UNITS OF EV/10**15 ATOMS/CM**2
C
      S=0.
C
      DO 60 I=1,NAT
      STP(I)=SN(I)*(8.462*Z1*A1*Z(I))/((A1+A(I))*SQRT(Z1**(2./3.)+Z(I)
     -**(2./3.)))
      S=S+STP(I)*DENSTRG(I)
   60 CONTINUE
C
      S=S*1.E-21
      RETURN
      END
C
C**********************FUNCTION ACOEF(I,Z)***************************
C
      FUNCTION ACOEF(I,Z)
C
C   FUNCTION ACOEF CONTAINS ZIEGLER STOPPING POWER COEFFICIENTS
C   FOR PROTONS
C
```

```
      DIMENSION A(12,36),ZB(5),AA(12,5)
C
      DATA ((A(I,J),I=1,12),J=1,14)/1.262,1.44,242.6,1.2E4,.1159,
     -.0005099,5.436E4,
     --5.052,2.049,-.3044,.01966,-.0004659,1.229,1.397,484.5,5873.,
     -.05225,.00102,2.451E4,-2.158,.8278,-.1172,.007259,-.000166,
     -1.411,1.6,725.6,3013.,.04578,.00153,2.147E4,-.5831,.562,-.1183,
     -.009298,-.0002498,2.248,2.59,966.,153.8,.03475,.002039,1.63E4,.2779
     -,.1745,-.05684,.005155,-.0001488,2.474,2.815,1206.,1060.,.02855
     -,.002549,1.345E4,-2.445,1.283,-.2205,.0156,-.000393,2.631,2.989,
     -1445.,957.2,.02819,.003059,1.322E4,-4.38,2.044,-.3283,.02221,
     --.0005417,2.954,3.35,1683.,1900.,.02513,.003569,1.179E4,-5.054,
     -2.325,-.3713,.02506,-.0006109,2.652,3.,1920.,2000.,.0223,.004079,
     -1.046E4,-6.734,3.019,-.4748,.03171,-.0007669,2.085,2.352,2157.,
     -2634.,.01816,.004589,8517.,-5.571,2.449,-.3781,.02483,-.0005919,
     -1.951,2.199,2393.,2699.,.01568,.005099,7353.,-4.408,1.879,-.2814,
     -.01796,-.0004168,2.542,2.869,2628.,1854.,.01472,.005609,6905.,
     --4.959,2.073,-.3054,.01921,-.0004403,3.792,4.293,2862.,1009.,
     -.01397,.006118,6551.,-5.51,2.266,-.3295,.02047,-.0004637,
     -4.154,4.739,2766.,164.5,.02023,.006628,6309.,-6.061,2.46,-.3535,
     -.02173,-.0004871,4.15,4.7,3329.,550.,.01321,.007138,6194.,-6.294,
     -2.538,-.3628,.0222,-.0004956/
      DATA ((A(I,J),I=1,12),J=15,28)/3.232,3.647,3561.,1560.,.01267,
     -.007648,5942.,-6.527,2.616,-.3721,.02267,-.000504,3.447,3.891,
     -3792.,1219.,.01211,.008158,5678.,-6.761,2.694,-.3814,.02314,
     --.0005125,5.047,5.714,4023.,878.6,.01178,.008668,5524.,-6.994,2.773
     -,-.3907,.02361,-.0005209,5.731,6.5,4253.,530.,.01123,.009178,
     -5268.,-7.227,2.851,-.4,.02407,-.0005294,
     -5.151,5.833,4482.,545.7,.01129,.009687,5295.,-7.44,2.923,
     --.4094,.02462,-.0005411,5.521,6.252,4710.,553.3,.01112,.0102,
     -5214.,-7.653,2.995,-.4187,.02516,-.0005529,
     -5.201,5.884,4938.,560.9,.009995,.01071,4688.,-8.012,3.123,-.435,
     -.02605,-.0005707,4.862,5.496,5165.,568.5,.009474,.01122,4443.,
     --8.371,3.251,-.4513,.02694,-.0005886,4.48,5.055,5391.,952.3,
     -.009117,.01173,4276.,-8.731,3.379,-.4676,.02783,-.0006064,
     -3.983,4.489,5616.,1336.,.008413,.01224,3946.,-9.09,3.507,-.4838,
     -.02872,-.0006243,3.469,3.907,5725.,1461.,.008829,.01275,3785.,
     --9.449,3.635,-.5001,.02961,-.0006421,3.519,3.963,6065.,1243.,
     -.007782,.01326,3650.,-9.809,3.763,-.5164,.0305,-.00066,3.14,
     -3.535,6288.,1372.,.007361,.01377,3453.,-10.17,3.891,-.5327,
     -.03139,-.0006779,3.553,4.004,6205.,555.1,.008763,.01428,
     -3297.,-10.53,4.019,-.549,.03229,-.0006957/
      DATA ((A(I,J),I=1,12),J=29,36)/3.696,4.175,4673.,
     -387.8,.02188,.01479,3174.,-11.18,4.252,-.5791,.03399,-.0007314,
     -4.21,4.75,6953.,295.2,.006809,.0153,3194.,-11.57,4.394,-.598,
     -.03506,-.0007537,5.041,5.697,7173.,202.6,.006725,.01581,
     -3154.,-11.95,4.537,-.6169,.03613,-.0007759,5.554,6.3,6496.,
     -110.,.009689,.01632,3097.,-12.34,4.68,-.6358,.03721,-.0007981,
     -5.323,6.012,7611.,292.5,.006447,.01683,3024.,-12.72,4.823,
     --.6547,.03828,-.0008203,5.874,6.656,7395.,117.5,.007684,
     -.01734,3006.,-13.11,4.965,-.6735,.03935,-.0008425,5.611,
     -6.335,8046.,365.2,.006244,.01785,2928.,-13.4,5.083,-.6906,
     -.04042,-.0008675,6.411,7.25,8262.,220.,.006087,.01836,2855.,
     --13.69,5.2,-.7076,.0415,-.0008925/
      DATA ZB/40.,47.,72.,82.,92./
      DATA AA/6.734,7.603,9120.,405.2,.005765,.02039,
     -2704.,-14.59,5.463,-.7333,.04242,-.0008998,
     -5.623,6.354,7160.,337.6,.01394,.02396,
     -2193.,-18.39,6.86,-.9211,.05346,-.00114,
     -5.028,5.657,1.555E4,440.8,.003195,.03671,
     -1499.,-15.18,5.22,-.6353,.03286,-.0006133,
     -5.319,5.982,1.74E4,586.3,.002871,.04181,
     -1347.,-19.48,6.857,-.8678,.04747,-.0009548,
     -7.29,8.204,1.918E4,586.3,.002573,.04691,
     -1207.,-20.18,6.995,-.8757,.04753,-.0009508/
C
```

```
      J=Z
      IF(Z.LE.36.) THEN
      ACOEF=A(I,J)
      RETURN
      ELSE IF(Z.LT.ZB(1)) THEN
      ACOEF=A(I,36)+(AA(I,1)-A(I,36))*(Z-ZB(1))/(ZB(1)-36.)
      RETURN
      ELSE
C
      DO 3 II=1,5
      IIZ=II
      IF(J.LT.ZB(II)) GO TO 4
    3 CONTINUE
C
    4 CONTINUE
      ACOEF=AA(I,IIZ-1)+(AA(I,IIZ)-AA(I,IIZ-1))*
     -(Z-ZB(IIZ-1))/(ZB(IIZ)-ZB(IIZ-1))
      RETURN
      END IF
      END
C
C*********************FUNCTION CCOEF(I,Z)*****************************
C
      FUNCTION CCOEF(I,Z)
C
C   FUNCTION CCOEF CONTAINS ZIEGLER STOPPING POWER COEFFICIENTS
C   FOR ALPHA PARTICLES
C   FOR A DISCUSSION SEE HEALTH PHYS. VOL.46, P.1101,1984
C
      DIMENSION C(9,37),CC(9,6),ZB(6)
C
C   ARRAY C IS BASED ON THE FOLLOWING LOW ENERGY GAS/SOLID CHOICES
C   G=GAS,S=SOLID
C   H-G,HE-G,LI-S,BE-S,B-S,C-G,N-G,O-G,F-G,NE-G,NA-G,MG-S,AL-S,
C   SI-S,P-S,S-G,CL-G,AR-G,K-S,CA-S,SC-S,TI-S,V-S,CR-S,MN-S,FE-S,
C   CO-S,NI-S,CU-S,ZN-S,GA-S,GE-S,AS-S,SE-S,BR-G,KR-G,RB-S
C   ARRAY ZB IS BASED ON
C   ZR-S,MO-S,AG-S,AU-S,PB-S,U-S
C
      DATA ((C(I,J),I=1,9),J=1,18)/
     -.39,.63,4.17,85.55,19.55,2.371,.5462,-.07932,-.006853,
     -.58,.59,6.3,130.,44.07,2.809,.4847,-.08756,-.007281,
     -1.42,.49,12.25,32.,9.161,3.095,.4434,-.09259,-.007459,
     -2.206,.51,15.32,.25,8.995,3.28,.4188,-.09564,-.007604,
     -3.691,.4128,18.48,50.72,9.,3.426,.4,-.09796,-.007715,
     -3.47,.4485,22.37,36.41,7.993,3.588,.3921,-.09935,-.007804,
     -2.,.548,29.82,18.11,4.37,3.759,.4094,-.09646,-.007661,
     -2.717,.4858,32.88,25.88,4.336,3.782,.3734,-.1011,-.007874,
     -2.616,.4708,41.2,28.07,2.458,3.816,.3504,-.1046,-.008074,
     -2.303,.4861,37.01,37.96,5.092,3.863,.3342,-.1072,-.008231,
     -13.03,.2685,35.65,44.18,9.175,3.898,.3191,-.1086,-.008271,
     -4.3,.47,34.3,3.3,12.74,3.961,.314,-.1091,-.008297,
     -2.5,.625,45.7,.1,4.359,4.024,.3113,-.1093,-.008306,
     -2.1,.65,49.34,1.788,4.133,4.077,.3074,-.1089,-.008219,
     -1.729,.6562,53.41,2.405,3.845,4.124,.3023,-.1094,-.00824,
     -3.116,.5988,53.71,5.632,4.536,4.164,.2964,-.1101,-.008267,
     -3.365,.571,63.67,6.182,2.969,4.21,.2936,-.1103,-.00827,
     -2.291,.6284,73.88,4.478,2.066,4.261,.2994,-.1085,-.008145/
      DATA ((C(I,J),I=1,9),J=19,37)/8.554,.3817,
     -83.61,11.84,1.875,4.3,.2903,-.1103,-.008259,
     -6.297,.4622,65.39,10.14,5.036,4.344,.2897,-.1102,-.008245,
     -5.307,.4918,61.74,12.4,6.665,4.327,.2707,-.1127,-.00837,
     -4.71,.5087,65.28,8.806,5.948,4.34,.2618,-.1138,-.00842,
     -6.151,.4524,83.,18.31,2.71,4.361,.2559,-.1145,-.008447,
     -6.57,.4322,84.76,15.33,2.779,4.349,.24,-.1166,-.00855,
     -5.738,.4492,84.61,14.18,3.101,4.362,.2327,-.1174,-.008588,
```

```
     -5.013,.4707,85.58,16.55,3.211,4.375,.2253,-.1185,-.008648,
     -4.32,.4947,76.14,10.85,5.441,4.362,.2069,-.1214,.008815,
     -4.652,.4571,80.73,22.,4.952,4.346,.1857,-.1249,-.009021,
     -3.114,.5236,76.67,7.62,6.385,4.355,.18,-.1255,-.009045,
     -3.114,.5236,76.67,7.62,7.502,4.389,.1806,-.1253,-.009028,
     -3.114,.5236,76.67,7.62,8.514,4.407,.1759,-.1258,-.009054,
     -5.746,.4662,79.24,1.185,7.993,4.419,.1694,-.1267,-.009094,
     -2.792,.6346,106.1,.2986,2.311,4.412,.1545,-.1289,-.009202,
     -4.667,.5095,124.3,2.102,1.667,4.419,.1448,-.1303,-.009269,
     -1.65,.7,148.1,1.47,.9686,4.436,.1443,-.1299,-.009229,
     -1.413,.7377,147.9,1.466,1.016,4.478,.1608,-.1262,-.008983,
     -11.72,.3826,102.8,9.231,4.371,4.489,.1517,-.1278,-.009078/
      DATA ZB/40.,42.,47.,79.,82.,92./
      DATA CC/10.99,.41,163.9,7.1,1.052,4.548,.1572,-.1256,-.008901,
     -9.276,.418,157.1,8.038,1.29,4.548,.1485,-.1259,-.008889,
     -5.6,.49,130.,10.,2.844,4.577,.13,-.1285,-.009067,
     -3.223,.5883,232.7,2.954,1.05,4.564,-.027,-.1471,-.009852,
     -6.18,.52,170.,4.,3.224,4.608,-.02886,-.1485,-.00999,
     -5.218,.5828,245.,3.838,1.25,4.751,.004244,-.1419,-.009576/
C
      J=Z
      IF(Z.LE.37.) THEN
      CCOEF=C(I,J)
      RETURN
      ELSE IF(Z.LT.ZB(1)) THEN
      CCOEF=C(I,37)+(CC(I,1)-C(I,37))*(Z-ZB(1))/(ZB(1)-37.)
      RETURN
      ELSE
C
      DO 3 II=1,6
      IIZ=II
      IF(J.LT.ZB(II)) GO TO 4
    3 CONTINUE
C
    4 CONTINUE
      CCOEF=CC(I,IIZ-1)+(CC(I,IIZ)-CC(I,IIZ-1))*
     -(Z-ZB(IIZ-1))/(ZB(IIZ)-ZB(IIZ-1))
      RETURN
      END IF
      END
C
C ***********************************************************
C
      SUBROUTINE MGAUSS(A,B,N,SUM,FUNC,FOFX,NUMBER)
C
C   SUBROUTINE MGAUSS USES A GAUSS-LEGENDRE QUADRATURE FORMULA TO
C   PERFORM A 5 POINTS PER SUBINTERVAL NUMERICAL INTEGRATION
C
C   A=LOWER LIMIT
C   B=UPPER LIMIT
C   N=NUMBER OF INTERVALS
C   SUM=RESULT OF INTEGRATION
C   FUNC=NAME OF SUBROUTINE THAT GIVES INTEGRAND
C   FOFX=INTEGRAND
C   NUMBER=NUMBER OF INTEGRALS BEING DONE
C
      DIMENSION U(5),R(5),SUM(1),FOFX(1)
C
      DO 10 LL=1,NUMBER
   10 SUM(LL)=0.0
C
      IF(A.EQ.B) RETURN
C
C   U(I) ARE THE ZEROS OF LEGENDRE POLYNOMIAL OF DEG. 5
C
      U(1)=.42556283050918
```

```
      U(2)=.28330230293537
      U(3)=.160295215850488
      U(4)=.067468316655508
      U(5)=.013046735741414
C
C   R(I) ARE THE WEIGHTS SUCH THAT THE INTEGRATION IS EXACT
C
      R(1)=.147762112357376
      R(2)=.13463335965499
      R(3)=.109543181257991
      R(4)=.074725674575290
      R(5)=.033335672154344
      FINE=N
      DELTA=FINE/(B-A)
C
C   FIND INITIAL VALUE FOR INTERVAL
C
      DO 30 K=1,N
      XI =K-1
      FINE=A+XI/DELTA
C
C   EVALUATE FUNCTION AT 5 POINTS PER INTERVAL
C
      DO 20 II=1,5
      UU=U(II)/DELTA+FINE
      CALL FUNC (UU,FOFX)
C
C   ADD TO SUM FOR EACH FUNCTION
C
      DO 20 NN=1,NUMBER
   20 SUM(NN)=R(II)*FOFX(NN)+SUM(NN)
C
      DO 30 II=1,5
      UU=(1.0-U(II))/DELTA+FINE
      CALL FUNC (UU,FOFX)
C
      DO 30 NN=1,NUMBER
   30 SUM(NN)=R(II)*FOFX(NN)+SUM(NN)
C
      DO 40 I=1,NUMBER
   40 SUM(I)=SUM(I)/DELTA
      RETURN
      END
C
C*****SUBROUTINE IUNI(NMAX,N,X,NTAB,Y,IORDER,X0,Y0,IPT,IERR)********
C
      SUBROUTINE IUNI(NMAX,N,X,NTAB,Y,IORDER,X0,Y0,IPT,IERR)
C
C   SUBROUTINE IUNI USES A 1-ST OR 2-ND ORDER LAGRANGIAN INTERPOLATION
C   FOR A MONOVARIATE FUNCTION Y=F(X)
C
C   TAKEN FROM LANGLEY MATH. LIBRARY
C
      DIMENSION X(NMAX),Y(NMAX,NTAB),Y0(NTAB)
C
      NM1=N-1
      IERR=0
      J=1
      DELX=X(2)-X(1)
      IF (IORDER .EQ. 0) GO TO 10
      IF (N.LT. 2) GO TO 20
      GO TO 50
   10 IERR=-1
      GO TO 30
   20 IERR=-2
C
```

```
   30 DO 40 NT=1,NTAB
      YO(NT)=Y(1,NT)
   40 CONTINUE
C
      RETURN
   50 IF (IPT .GT. -1) GO TO 65
      IF (DELX .EQ. 0) GO TO 190
      IF (N .EQ. 2) GO TO 65
C
      DO 60 J=2,NM1
      IF (DELX * (X(J+1)-X(J))) 190,190,60
   60 CONTINUE
C
   65 IF (IPT .LT. 1) IPT=1
      IF (IPT .GT. NM1) IPT=NM1
      IN= SIGN (1.0,DELX *( XO-X(IPT)))
   70 P= X(IPT) - XO
      IF (P* (X(IPT +1)- XO)) 90,180,80
   80 IPT =IPT +IN
      IF (IPT.GT.0 .AND. IPT .LT. N) GO TO 70
      IERR=-4
      IPT=IPT- IN
   90 IF (IORDER .GT. 1) GO TO 120
C
      DO 100 NT=1,NTAB
      YO(NT)=Y(IPT,NT)+((Y(IPT+1,NT)- Y(IPT,NT))*(XO-X(IPT)))/
     -(X(IPT+1)-X(IPT))
  100 CONTINUE
C
      IF (IERR .EQ. -4) IPT=IPT+IN
      RETURN
  120 IF (N .EQ. 2) GO TO 200
      IF (IPT .EQ. NM1) GO TO  140
      IF (IPT .EQ. 1) GO TO 130
      IF (DELX *(XO-X(IPT-1)).LT.DELX* (X(IPT+2)-XO)) GO TO 140
  130 L=IPT
      GO TO 150
  140 L=IPT -1
  150 V1=X(L)-XO
      V2=X(L+1)-XO
      V3=X(L+2)-XO
C
      DO 160 NT=1,NTAB
      YY1=(Y(L,NT) * V2 - Y(L+1,NT) * V1)/(X(L+1) - X(L))
      YY2=(Y(L+1,NT)*V3-Y(L+2,NT) *V2)/(X(L+2)-X(L+1))
  160 YO(NT)=(YY1*V3 -YY2*V1)/(X(L+2)-X(L))
C
      IF (IERR .EQ. -4) IPT=IPT + IN
      RETURN
  180 IF(P .NE. 0) IPT=IPT +1
C
      DO 185 NT=1,NTAB
      YO(NT)=Y(IPT,NT)
  185 CONTINUE
C
      RETURN
  190 IERR=J +1
      RETURN
  200 IERR=-3
      RETURN
      END
C
C*****SUBROUTINE IBI(NX,X,NY,Y,MAXF,F,IORDER,IPT,XO,YO,Z,IERR)******
C
      SUBROUTINE IBI(NX,X,NY,Y,MAXF,F,IORDER,IPT,XO,YO,Z,IERR)
C
```

```
C    SUBROUTINE IBI USES A 1-ST OR 2-ND ORDER LAGRANGIAN INTERPOLATION
C    FOR A BIVARIATE FUNCTION Z=F(X,Y)
C
C    TAKEN FROM LANGLEY MATH LIBRARY
C
C    REQUIRED ROUTINES QXZ114,QXZ115,QXZ116,QXZ117
C
      DIMENSION X(NX),Y(NY),F(MAXF,NY),P(3),S(3),IPT(2),IORDER(2)
C
      IF(IPT(1).NE.-1)GO TO 10
      IERR=-1
      IF(QXZ114(NX,X(1),IPT(1)).LT.0)RETURN
      IF(QXZ114(NY,Y(1),IPT(2)).LT.0)RETURN
      IPT(1)=1
      IPT(2)=1
C     IPT(1)=IPT(2)=1
   10 IERR=-4
      IF(IORDER(1).NE.1.AND.IORDER(1).NE.2)RETURN
      IF(IORDER(2).NE.1.AND.IORDER(2).NE.2)RETURN
      IERR=-2
      IF(NX-1.LT.IORDER(1))RETURN
      IF(NY-1.LT.IORDER(2))RETURN
      IERR=0
      CALL QXZ116(X0,X,IPT(1),NX,M,IERR)
      CALL QXZ116(Y0,Y,IPT(2),NY,N,IERR)
      CALL QXZ115(IPT(2),IORDER(2),Y0,KL,KR,Y,NY)
      KM=KL+1
      CALL QXZ117(Y0,Y(KR),Y(KM),Y(KL),P,IORDER(2))
      CALL QXZ115(IPT(1),IORDER(1),X0,LL,LR,X,NX)
      LM=LL+1
      CALL QXZ117(X0,X(LR),X(LM),X(LL),S,IORDER(1))
      Z=0.
      LP1=0
C
      DO 30 I=LL,LR
      LP1=LP1+1
      SUM=0.
      KP1=0
      DO 20 J=KL,KR
      KP1=KP1+1
   20 SUM=P(KP1)*F(I,J)+SUM
C
   30 Z=SUM*S(LP1)+Z
      IF(IERR.GE.0)RETURN
      IPT(1)=M
      IPT(2)=N
      RETURN
      END
C
C*****************FUNCTION QXZ114(M,VAR,IPOS)***********************
C
      FUNCTION QXZ114(M,VAR,IPOS)
C
C    FUNCTION QXZ114 CHECKS TO SEE IF SEQUENCE VAR IS INCREASING
C    TAKEN FROM LANGLEY MATH. LIBRARY
C
      DIMENSION VAR(M)
C
      QXZ114=1
      K=M-1
C
      DO 10 L=1,K
      N=L+1
      IF(VAR(N)-VAR(L).GT.0.)GO TO 10
      QXZ114=-1
      IPOS=N
```

```
   10 CONTINUE
C
      RETURN
      END
C
C***********SUBROUTINE QXZ115(IPT,IORDER,S,IL,IR,VAR,N)*************
C
      SUBROUTINE QXZ115(IPT,IORDER,S,IL,IR,VAR,N)
C
C   SUBROUTINE QXZ115 DECIDES THE INDICIES OF X,Y,Z ARRAY PASSES
C   TO ARRAY VAR
C
C   TAKEN FROM LANGLEY MATH. LIBRARY
C
      DIMENSION VAR(N)
C
      IF(S.LE.VAR(2))GO TO 10
      IF(S.GE.VAR(N-1))GO TO 20
      IL=IPT
      IF(S.LE.VAR(IPT))IL=IL-1
      IF(IORDER.LT.2)GO TO 30
      IF(ABS(S-VAR(IL-1)).LT.ABS(S-VAR(IL+2)))IL=IL-1
      GO TO 30
   10 IL=1
      GO TO 30
   20 IL=N-IORDER
   30 IR=IL+IORDER
      RETURN
      END
C
C***********SUBROUTINE QXZ116(S,VAR,IPOS,NBND,IF,IERR)**************
C
      SUBROUTINE QXZ116(S,VAR,IPOS,NBND,IF,IERR)
C
C   SUBROUTINE QXZ116 DECIDES THE INDEX OF X,Y,Z ARRAY(VAR)
C   NEAREST TO X0,Y0,Z0
C
C   TAKEN FROM LANGLEY MATH. LIBRARY
C
      DIMENSION VAR(NBND)
C
      LOGICAL ABOVE,BELOW
C
      ABOVE=.FALSE.
      BELOW=.FALSE.
C     ABOVE=BELOW=.FALSE.
      IF(IPOS.EQ.0)IPOS=1
   10 IF(S-VAR(IPOS))20,50,30
   20 ABOVE=.TRUE.
      IUP=IPOS
      IF(ABOVE.AND.BELOW)GO TO 40
      IPOS=IPOS-1
      IF(IPOS.GE.1)GO TO 10
      IERR=-3
      IPOS=1
      IF=0
      RETURN
   30 BELOW=.TRUE.
      LOW=IPOS
      IF(ABOVE.AND.BELOW)GO TO 40
      IPOS=IPOS+1
      IF(IPOS.LE.NBND)GO TO 10
      IERR=-3
      IF=NBND
      IPOS=NBND
      RETURN
```

```
   40 IPOS=LOW
      IF(ABS(S-VAR(IUP)).LT.ABS(S-VAR(LOW)))IPOS=IUP
   50 IF=IPOS
      RETURN
      END
C
C***********SUBROUTINE QXZ117(S,VAR1,VAR2,VAR3,Q,IORDER)************
C
      SUBROUTINE QXZ117(S,VAR1,VAR2,VAR3,Q,IORDER)
C
C    SUBROUTINE QXZ117 CALCULATES THE LAGRANGIAN COEFF. FOR
C    1-ST OR 2-ND ORDER INTERPOLATION
C
C    TAKEN FROM LANGLEY MATH. LIBRARY
C
      DIMENSION Q(3)
C
      IF(IORDER.EQ.2)GO TO 10
      T1=VAR3-VAR1
      T2=S-VAR1
      T3=S-VAR3
      Q(1)=T2/T1
      Q(2)=-T3/T1
      RETURN
   10 T1=S-VAR2
      T2=S-VAR1
      T3=S-VAR3
      Q(1)=T1*T2
      Q(2)=T3*T2
      Q(3)=T3*T1
      T1=VAR3-VAR2
      T2=VAR3-VAR1
      T3=VAR2-VAR1
      Q(1)=Q(1)/(T1*T2)
      Q(2)=Q(2)/(-T1*T3)
      Q(3)=Q(3)/(T2*T3)
      RETURN
      END
C
C*******************************************************
C
      FUNCTION TEXP(X)
C
      IF(X.LT.-88.) THEN
      TEXP=EXP(-88.)
      RETURN
      ELSE IF(X.GT.88.) THEN
      TEXP=EXP(88.)
      RETURN
      ELSE
      TEXP=EXP(X)
      END IF
      RETURN
      END
C
C *******************************************************
C
      SUBROUTINE SUTPAR(NX,NY,X,Y,NZ,Z,IENDSW,ENDX1,ENDXN,ENDY1,
     $                  ENDYN,ENDXY,SIGMA,M,XX,YY,IW,ZZ,WK,
     $                  IERR)
C
C    SUBROUTINE SUTPAR INTERPOLATE A BIVARIATE SPLINE UNDER TENSION
C    AT A VECTOR OF POINTS, RETURNING THE FUNCTION VALUE, THE TWO FIRST
C    PARTIAL DERIVATIVES, AND THE THREE SECOND PARTIAL DERIVATIVES
C
C    TAKEN FROM LANGLEY MATH. LIBRARY
```

```
C
CC   REQUIRED ROUTINES - QXZ060,QXZ063,QXZ062,QXZ119,QXZ111,QXZ061,QXZ112,
CC
CC   FORMAL PARAMETERS
C
      INTEGER IENDSW(8),IERR,IW,M,NX,NY,NZ
C
      REAL ENDX1(NY),ENDXN(NY),ENDY1(NX),ENDYN(NX),ENDXY(4),SIGMA,WK(*)
      REAL X(NX),XX(M),Y(NY),YY(M),Z(NZ,NY),ZZ(6,M)
C
CC   INTERNAL VARIABLES
C
      INTEGER INDEX,K
C
      REAL ABSIGMA,DELX,DELY,ONE,SIGMAX,SIGMAY,ZERO
C
CC   INITIALIZE CONSTANTS
C
      DATA ZERO,ONE/0.0E0,1.0E0/
C
CC   INITIALIZE  IERR  AND CHECK ERROR RETURNS
C
      IERR = 1
      IF (NX .LE. 1) GO TO 9000
      IF(M.LE.0)  GO TO 9000
      IF (NY .LE. 1) GO TO 9000
      IERR = 2
      DELX = (X(NX) - X(1))/(NX - 1)
      IF (DELX .LE. ZERO) GO TO 9000
      DELY = (Y(NY) - Y(1))/(NY - 1)
      IF (DELY .LE. ZERO) GO TO 9000
      IERR = 0
C
CC   DENORMALIZE THE TENSION FACTOR  SIGMA  FOR  X  AND  Y
CC   IF SIGMA IS LESS THAN 1.E-30, IT IS REDEFINED TO ZERO
CC   (WITH NO SIGNIFICANT EFFECT) TO AVOID UNDERFLOW.
C
      ABSIGMA = ABS(SIGMA)
      IF (ABSIGMA .LT. 1.E-30) ABSIGMA = 0.0
      SIGMAX = ABSIGMA / DELX
      SIGMAY = ABSIGMA / DELY
C
CC   BRANCH IF  QXZ119  HAS ALREADY BEEN CALLED
C
      IF (IW .EQ. 1) GO TO 30
      INDEX = NX*NY*3 + 1
      IF (IENDSW(3) .NE. 2) GO TO 10
      IENDSW(5) = 0
      ENDXY(1) = ZERO
      IENDSW(6) = 0
      ENDXY(2) = ZERO
   10 IF (IENDSW(4) .NE. 2) GO TO 20
      IENDSW(7) = 0
      ENDXY(3) = ZERO
      IENDSW(8) = 0
      ENDXY(4) = ZERO
   20 CALL QXZ119(NX,NY,X,Y,NZ,Z,IENDSW,ENDX1,ENDXN,ENDY1,ENDYN,
     $             ENDXY(1),ENDXY(2),ENDXY(3),ENDXY(4),DELX,SIGMAX,
     $             DELY,SIGMAY,0,0,WK,WK(INDEX),IERR)
      IF (IERR .NE. 0) GO TO 9000
C
CC   IW  MUST BE SET TO 1 FOR OUTPUT
C
      IW = 1
C
   30 CONTINUE
```

```
C
C   BEGIN LOOP FOR INTERPOLATION AND DERIVATIVE EVALUATION
C
      DO 90 K=1,M
      CALL QXZ111(XX(K),YY(K),ZZ(1,K),ZZ(2,K),ZZ(3,K),ZZ(5,K),
     *            ZZ(4,K),ZZ(6,K),NX,NY,X,Y,Z,NZ,WK,SIGMA,IERR)
   90 CONTINUE
      RETURN
 9000 IF(IERR .EQ. 1) IERR = -2
      IF(IERR .EQ. 2) IERR = -1
      RETURN
      END
C
C ***************************************************
C
      SUBROUTINE QXZ060 (DEL1,DEL2,SIGMA,C1,C2,C3,N)
C
C   SUBROUTINE QXZ060 DETERMINE COEFFICIENTS  C1 ,  C2 , AND  C3
C   USED TO DETERMINE ENDPOINT SLOPES.  GIVEN THE
C   POINTS  (X1,Y1) ,  (X2,Y2) , AND  (X3,Y3) ,
C   THE ENDPOINT SLOPE IS  C1*Y1 + C2*Y2 + C3*Y3 *
C   IF  N  > 2 AND  C1*Y1 + C2*Y2  IF  N  = 2.
C
C   FORMAL PARAMETERS
C
      INTEGER N
C
      REAL C1,C2,C3,DEL1,DEL2,SIGMA
C
C   INTERNAL VARIABLES
C
      REAL COSHM1,DEL,DENOM,ONE,SIN,TWO,ZERO
C
C   INITIALIZE CONSTANTS
C
      DATA ZERO,ONE,TWO/0.0E0,1.0E0,2.0E0/
C
C   TEST WHETHER  N  = 2
C
      IF (N .EQ. 2) GO TO 20
      IF (ABS(SIGMA*DEL2) .GT. 740.0) GO TO 30
C
C   INITIALIZE THE VARIABLE  DEL
C
      DEL = DEL2 - DEL1
C
C   TEST WHETHER  SIGMA  = ZERO
C
      IF (SIGMA .NE. ZERO) GO TO 10
C
C   THIS CODE IS FOR A STANDARD CUBIC SPLINE
C
      C1 = -(DEL1+DEL2)/(DEL1*DEL2)
      C2 = DEL2/(DEL1*DEL)
      C3 = -DEL1/(DEL2*DEL)
      GO TO 9000
C
C   THIS CODE IS FOR A SPLINE UNDER TENSION
C
   10 CALL QXZ062(DUMMY,COSHM1,SIGMA*DEL1,1)
      CALL QXZ062(DUMMY,COSHM2,SIGMA*DEL2,1)
      SIN = TWO * SINH(SIGMA*(DEL2+DEL1)/TWO) * SINH(SIGMA*DEL/TWO)
      DENOM = ONE / (COSHM1*DEL - DEL1*SIN)
      C1 = SIN * DENOM
      C2 = -COSHM2 * DENOM
      C3 = COSHM1 * DENOM
```

```
      GO TO 9000
C
C   TO AVOID OVERFLOW IF SIGMA*DEL2 IS TOO LARGE
C
   30 C3 = 0.0
C
C   TWO COEFFICIENTS
C
   20 C1 = -ONE/DEL1
      C2 = -C1
 9000 RETURN
      END
C
C *******************************************************
C
      SUBROUTINE QXZ061(DIAG,SDIAG,SIGMA,DEL)
C
C   SUBROUTINE QXZ061 COMPUTE THE DIAGONAL AND SUPERDIAGONAL TERMS
C   OF THE TRIDIAGONAL LINEAR SYSTEM ASSOCIATED
C   WITH SPLINE UNDER TENSION INTERPOLATION.
C
C   FORMAL PARAMETERS
C
      REAL DEL,DIAG,SDIAG,SIGMA
C
C   INTERNAL VARIABLES
C
      REAL COSHM,DENOM,SIGDEL,SINHM
      REAL*8 SIXTH,THIRD
C
C   INITIALIZE CONSTANTS
C
      DATA ZERO,SIXTH,THIRD/
     $0.0E0,'17155252525252525253'O,'17165252525252525253'O/
C
C   TEST WHETHER  SIGMA  IS ZERO
C
      IF (SIGMA .NE. ZERO) GO TO 10
C
C   THIS CODE IS FOR THE STANDARD CUBIC SPLINE
C
      DIAG = DEL*THIRD
      SDIAG = DEL*SIXTH
      GO TO 9000
C
C   THIS CODE IS FOR A SPLINE UNDER NON-ZERO TENSION
C
   10 SIGDEL = SIGMA*DEL
      CALL QXZ062(SINHM,COSHM,SIGDEL,0)
      DENOM = SIGMA*SIGDEL*(1.+SINHM)
      DIAG = (COSHM-SINHM)/DENOM
      SDIAG = SINHM/DENOM
C
 9000 RETURN
      END
C
C *******************************************************
C
      SUBROUTINE QXZ062 (SINHM,COSHM,X,ISW)
C
C   SUBROUTINE QXZ062 RETURNS APPROXIMATIONS TO
C   SINHM(X)=SINH(X)/X-1
C   COSHM(X)=COSH(X)-1
C   COSHMM(X)=(COSH(X)-1-X*X/2)/(X*X)
C   WITHOUT RELATIVE ERROR LESS THAN 4.0E-14.
C
```

```
      INTEGER ISW
      REAL SINHM,COSHM,X
C
      DATA SP14/.22758166097634BE-7/,
     *     SP13/.612189863171694E-5/,
     *     SP12/.7153147592112O9E-3/,
     *     SP11/.398088289992973E-1/,
     *     SQ12/.2063827014137 25E-3/,
     *     SQ11/-.611470260009508E-1/,
     *     SQ10/.599999999999986E+1/
      DATA SP25/.129094158037272E-9/,
     *     SP24/.4737318231016 66E-7/,
     *     SP23/.8492134555984 55E-5/,
     *     SP22/.8332648033272 42E-3/,
     *     SP21/.4250241428132 26E-1/,
     *     SQ22/.1060085157448 21E-3/,
     *     SQ21/-.4498551695125 05E-1/,
     *     SQ20/.6000000002686 19E+1/
      DATA SP35/.1551939458649 42E-9/,
     *     SP34/.5115294516687 37E-7/,
     *     SP33/.8847756357767 84E-5/,
     *     SP32/.8504476176913 92E-3/,
     *     SP31/.4288881487917 77E-1/,
     *     SQ32/.9331288310616 10E-4/,
     *     SQ31/-.4266775705385 07E-1/,
     *     SQ30/.6000001450864 89E+1/
      DATA SP45/.1880706320583 31E-9/,
     *     SP44/.5457928177141 92E-7/,
     *     SP43/.9201195357952 22E-5/,
     *     SP42/.8665593916729 85E-3/,
     *     SP41/.4325352349608 58E-1/,
     *     SQ42/.8248917488206 70E-4/,
     *     SQ41/-.4049388416722 62E-1/,
     *     SQ40/.6000050062838 34E+1/
      DATA CP5/.5522006145847 44E-9/,
     *     CP4/.1816669236209 44E-6/,
     *     CP3/.2705401258465 25E-4/,
     *     CP2/.2062707195039 34E-2/,
     *     CP1/.7444372055690 40E-1/,
     *     CQ2/.5146096386426 89E-4/,
     *     CQ1/-.1777922555283 82E-1/,
     *     CQ0/.2000000000000 00E+1/
      DATA ZP4/.6644188058768 35E-8/,
     *     ZP3/.2182745356863 85E-5/,
     *     ZP2/.3248510593271 61E-3/,
     *     ZP1/.2445151501742 58E-1/,
     *     ZQ2/.6161657823066 21E-3/,
     *     ZQ1/-.2131636395794 25E0/,
     *     ZQ0/.2400000000000 00E+2/
C
      AX = ABS(X)
      IF (ISW .GE. 0) GO TO 5
C
C     SINHM APPROXIMATION
C
      IF (AX .GT. 3.9) GO TO 2
      XS = AX*AX
      IF (AX .GT. 2.2) GO TO 1
C
C     SINHM APPROXIMATION ON (0.,2.2)
C
      SINHM = XS*((((SP14*XS+SP13)*XS+SP12)*XS+SP11)*XS+1.)/
     .                ((SQ12*XS+SQ11)*XS+SQ10)
      RETURN
C
C     SINHM APPROXIMATION ON (2.2,3.9)
```

```
C
    1 SINHM = XS*((((SP25*XS+SP24)*XS+SP23)*XS+SP22)*XS+SP21)
     .         *XS+1.)/((SQ22*XS+SQ21)*XS+SQ20)
       RETURN
    2 IF (AX .GT. 5.1) GO TO 3
C
C   SINHM APPROXIMATION ON (3.9,5.1)
C
       XS = AX*AX
       SINHM = XS*((((SP35*XS+SP34)*XS+SP33)*XS+SP32)*XS+SP31)
     .         *XS+1.)/((SQ32*XS+SQ31)*XS+SQ30)
       RETURN
    3 IF (AX .GT. 6.1) GO TO 4
C
C   SINHM APPROXIMATION ON (5.1,6.1)
C
       XS = AX*AX
       SINHM = XS*((((SP45*XS+SP44)*XS+SP43)*XS+SP42)*XS+SP41)
     .         *XS+1.)/((SQ42*XS+SQ41)*XS+SQ40)
       RETURN
C
C   SINHM APPROXIMATION ABOVE 6.1
C
    4 EXPX = EXP(AX)
       SINHM = (EXPX-1./EXPX)/(AX+AX)-1.
       RETURN
C
C   COSHM AND (POSSIBLY) SINHM APPROXIMATION
C
    5 IF (ISW .GE. 2) GO TO 7
       IF (AX .GT. 2.2) GO TO 6
       XS = AX*AX
       COSHM = XS*((((CP5*XS+CP4)*XS+CP3)*XS+CP2)*XS+CP1)
     .         *XS+1.)/((CQ2*XS+CQ1)*XS+CQ0)
       IF (ISW .EQ. 0) SINHM = XS*((((SP14*XS+SP13)*XS+SP12)
     .           *XS+SP11)*XS+1.)/((SQ12*XS+SQ11)*XS+SQ10)
       RETURN
    6 EXPX = EXP(AX)
       COSHM = (EXPX+1./EXPX)/2.-1.
       IF (ISW .EQ. 0) SINHM = (EXPX-1./EXPX)/(AX+AX)-1.
       RETURN
C
C   COSHMM AND (POSSIBLY) SINHM APPROXIMATION
C
    7 XS = AX*AX
       IF (AX .GT. 2.2) GO TO 8
       COSHM = XS*((((ZP4*XS+ZP3)*XS+ZP2)*XS+ZP1)*XS+1.)/
     .                 ((ZQ2*XS+ZQ1)*XS+ZQ0)
       IF (ISW .EQ. 3) SINHM = XS*((((SP14*XS+SP13)*XS+SP12)
     .           *XS+SP11)*XS+1.)/((SQ12*XS+SQ11)*XS+SQ10)
       RETURN
    8 EXPX = EXP(AX)
       COSHM = ((EXPX+1./EXPX-XS)/2.-1.)/XS
       IF (ISW .EQ. 3) SINHM = (EXPX-1./EXPX)/(AX+AX)-1.
       RETURN
       END
C
C ***************************************************
C
       INTEGER FUNCTION QXZ063 (T,X,N)
C
C   FUNCTION QXZ063 DETERMINES THE INDEX OF THE INT
C   (DETERMINED) BY A GIVEN INCREASING SEQUENCE)
C   WHICH A GIVEN VALUE LIES.
C
       INTEGER N
```

```
      REAL T,X(N)
C
      SAVE I
      DATA I /1/
C
      TT = T
C
C   CHECK FOR ILLEGAL I
C
      IF (I .GE. N) I = N/2
C
C   CHECK OLD INTERVAL AND EXTREMES
C
      IF (TT .LT. X(I)) THEN
      IF (TT .LE. X(2)) THEN
      I = 1
      QXZ063 = 1
      RETURN
      ELSE
      IL = 2
      IH = I
      END IF
      ELSE IF (TT .LE. X(I+1)) THEN
      QXZ063 = I
      RETURN
      ELSE IF (TT .GE. X(N-1)) THEN
      I = N-1
      QXZ063 = N-1
      RETURN
      ELSE
      IL = I+1
      IH = N-1
      END IF
C
C   BINARY SEARCH LOOP
C
    1 I = (IL+IH)/2
      IF (TT .LT. X(I)) THEN
      IH = I
      ELSE IF (TT .GT. X(I+1)) THEN
      IL = I+1
      ELSE
      QXZ063 = I
      RETURN
      END IF
      GO TO 1
      END
C
C ******************************************************
C
      SUBROUTINE QXZ111 (XX,YY,ZZ,ZX,ZY,ZXX,ZXY,ZYY,M,N,X,Y,
     *                   Z,IZ,ZP,SIGMA,IERR)
C
C   THIS SUBROUTINE EVALUATES THE FUNCTION VALUE, THE TWO
C   FIRST PARTIAL DERIVATIVES, AND THE THREE SECOND PARTIAL
C   DERIVATIVES OF A TENSOR PRODUCT SPLINE UNDER TENSION IN
C   TWO VARIABLES. THE SUBROUTINE QXZ119 SHOULD BE CALLED
C   EARLIER TO DETERMINE CERTAIN NECESSARY PARAMETERS.
C
      INTEGER M,N,IZ
      INTEGER QXZ063
      REAL XX,YY,ZZ,ZX,ZY,ZXX,ZXY,ZYY,X(M),Y(N),Z(IZ,N),
     *     ZP(M,N,3),SIGMA
C
      REAL CC(10),ZJ(2),XJ(2),XXJ(2),YYJ(2),XYYJ(2),XXYYJ(2)
C
```

```
C   DENORMALIZE TENSION FACTOR IN X AND Y DIRECTIONS
C
      ABSIGMA = ABS(SIGMA)
      IF(ABSIGMA .LT. 1.E-30) ABSIGMA = 0.0
      SIGMAX = ABSIGMA*FLOAT(M-1)/(X(M)-X(1))
      SIGMAY = ABSIGMA*FLOAT(N-1)/(Y(N)-Y(1))
C
C   DETERMINE X INTERVAL
C
      IM1 = QXZ063(XX,X(1),M)
      I = IM1+1
C
C   DETERMINE Y INTERVAL
C
      JM1 = QXZ063(YY,Y(1),N)
      J = JM1+1
C
C   OBTAIN X MESH DISTANCES
C
      DEL1 = XX-X(IM1)
      DEL2 = X(I)-XX
      DELS = X(I)-X(IM1)
C
C   SET ERROR CODE TO PREVENT OVERFLOW DUE TO EXCESSIVE EXTRAPOLATION.
C
      IERR = 4
      IF (SIGMAX*ABS(DEL1) .GT. 650.0 .OR. SIGMAX*ABS(DEL2) .GT. 650.0)
     *                                     GO TO 9000
      IERR = 0
C
C   OBTAIN COMMON COEFFICIENTS
C
      CC(1) = DEL2/DELS
      CC(2) = DEL1/DELS
      D = -1./DELS
      CC(5) = D
      CC(6) = -D
      IF (SIGMAX .NE. 0.) GO TO 1
C
C   OBTAIN COEFFICIENTS FOR CUBIC SPLINE
C
      CON = -DEL1*DEL2/(6.*DELS)
      CC(3) = (DEL2+DELS)*CON
      CC(4) = (DEL1+DELS)*CON
      CC(7) = -(2.*DEL2*DEL2-DEL1*(DEL2+DELS))/(6.*DELS)
      CC(8) = (2.*DEL1*DEL1-DEL2*(DEL1+DELS))/(6.*DELS)
      CC(9) = CC(1)
      CC(10) = CC(2)
      GO TO 2
C
C   OBTAIN COEFFICIENTS FOR SPLINE UNDER TENSION
C
    1 SIGDEL = SIGMAX*DELS
      CALL QXZ062(SINHM1,COSHM1,SIGMAX*DEL1   , 0)
      CALL QXZ062(SINHM2,COSHM2,SIGMAX*DEL2   , 0)
      CALL QXZ062(SINHMS,DUMMY ,SIGDEL        ,-1)
      SINHS = 1. + SINHMS
      CON = 1./(SIGMAX*SIGDEL*SINHS)
      CC(3) = CON*DEL2*(SINHM2-SINHMS)
      CC(4) = CON*DEL1*(SINHM1-SINHMS)
      CC(7) = -CON*(COSHM2-SINHMS)
      CC(8) = CON*(COSHM1-SINHMS)
      CC(9) = (DEL2*(1.+SINHM2))/(DELS*SINHS)
      CC(10) = (DEL1*(1.+SINHM1))/(DELS*SINHS)
C
C   INTERPOLATE IN X DIRECTION
```

```
C
    2 CALL QXZ112 (CC,Z(IM1,JM1),ZP(IM1,JM1,2),ZJ(1),XJ(1),
     *              XXJ(1))
      CALL QXZ112 (CC,Z(IM1,J),ZP(IM1,J,2),ZJ(2),XJ(2),
     *              XXJ(2))
      CALL QXZ112 (CC,ZP(IM1,JM1,1),ZP(IM1,JM1,3),YYJ(1),
     *              XYYJ(1),XXYYJ(1))
      CALL QXZ112 (CC,ZP(IM1,J,1),ZP(IM1,J,3),YYJ(2),
     *              XYYJ(2),XXYYJ(2))
C
C   OBTAIN Y MESH DISTANCES
C
      DEL1 = YY-Y(JM1)
      DEL2 = Y(J)-YY
      DELS = Y(J)-Y(JM1)
C
C   SET ERROR CODE TO PREVENT OVERFLOW DUE TO EXCESSIVE EXTRAPOLATION.
C
      IERR = 4
      IF (SIGMAY*ABS(DEL1) .GT. 650.0 .OR. SIGMAY*ABS(DEL2) .GT. 650.0)
     *                                     GO TO 9000
      IERR = 0
C
C   OBTAIN COMMON COEFFICIENTS
C
      CC(1) = DEL2/DELS
      CC(2) = DEL1/DELS
      D = -1./DELS
      CC(5) = D
      CC(6) = -D
      IF (SIGMAY .NE. 0.) GO TO 3
C
C   OBTAIN COEFFICIENTS FOR CUBIC SPLINE
C
      CON = -DEL1*DEL2/(6.*DELS)
      CC(3) = (DEL2+DELS)*CON
      CC(4) = (DEL1+DELS)*CON
      CC(7) = -(2.*DEL2*DEL2-DEL1*(DEL2+DELS))/(6.*DELS)
      CC(8) = (2.*DEL1*DEL1-DEL2*(DEL1+DELS))/(6.*DELS)
      CC(9) = CC(1)
      CC(10) = CC(2)
      GO TO 4
C
C   OBTAIN COEFFICIENTS FOR SPLINE UNDER TENSION
C
    3 SIGDEL = SIGMAY*DELS
      CALL QXZ062(SINHM1,COSHM1,SIGMAY*DEL1    , 0)
      CALL QXZ062(SINHM2,COSHM2,SIGMAY*DEL2    , 0)
      CALL QXZ062(SINHMS,DUMMY ,SIGDEL         ,-1)
      SINHS = 1.+SINHMS
      CON = 1./(SIGMAY*SIGDEL*SINHS)
      CC(3) = CON*DEL2*(SINHM2-SINHMS)
      CC(4) = CON*DEL1*(SINHM1-SINHMS)
      CC(7) = -CON*(COSHM2-SINHMS)
      CC(8) = CON*(COSHM1-SINHMS)
      CC(9) = (DEL2*(1.+SINHM2))/(DELS*SINHS)
      CC(10) = (DEL1*(1.+SINHM1))/(DELS*SINHS)
C
C   INTERPOLATE IN Y DIRECTION
C
    4 CALL QXZ112 (CC,ZJ,YYJ,ZZ,ZY,ZYY)
      CALL QXZ112 (CC,XJ,XYYJ,ZX,ZXY,DUMMY)
      ZXX = CC(1)*XXJ(1)+CC(2)*XXJ(2)+CC(3)*XXYYJ(1)+
     *      CC(4)*XXYYJ(2)
 9000 RETURN
      END
```

```
C
C ***************************************************
C
      SUBROUTINE QXZ112 (A,Y,YP,YY,YYP,YYPP)
C
      REAL A(10),Y(2),YP(2)
C
      YY = A(1)*Y(1)+A(2)*Y(2)+A(3)*YP(1)+A(4)*YP(2)
      YYP = A(5)*Y(1)+A(6)*Y(2)+A(7)*YP(1)+A(8)*YP(2)
      YYPP = A(9)*YP(1)+A(10)*YP(2)
      RETURN
      END
C
C ***********************************************************
C
      SUBROUTINE QXZ119(M,N,X,Y,IZ,Z,ISLPSW,ZX1,ZXM,ZY1,ZYN,ZXY11,ZXYM1,
     $                  ZXY1N,ZXYMN,DELX,SIGMAX,DELY,SIGMAY,IOPTX,IOPTY,
     $                  ZP,TEMP,IERR)
C
C   SUBROUTINE QXZ119 COMPUTE PARAMETERS NECESSARY TO BE ABLE TO
C   INTERPOLATE POINTS ON A BIVARIATE SPLINE
C   FUNCTION UNDER TENSION PASSING THROUGH A
C   RECTANGULAR GRID OF FUNCTIONAL VALUES.  SEE
C   STIBI (E1.X).
C
C   FORMAL PARAMETERS
C
      INTEGER IERR,IOPTX,IOPTY,ISLPSW(8),IZ,M,N
C
      REAL DELX,DELY,SIGMAX,SIGMAY,TEMP(*),X(M),Y(N),Z(IZ,N),ZP(M,N,3)
      REAL ZXM(N),ZX1(N),ZYN(M),ZY1(M),ZXYMN,ZXYM1,ZXY1N,ZXY11
C
C   INTERNAL PARAMETERS
C
      INTEGER I,IBAK,IBAKP1,IM1,IP1,J,JBAK,JBAKP1,JM1,JP1,MM1,MP1,NM1
      INTEGER NPI,NPM,NPMPJ,NP1
C
      REAL C1,C2,C3,DELI,DELXM,DELXMM,DELX1,DELX2,DELYN,DELYNM,DELY1
      REAL DELY2,DEL1,DEL2,DIAGI,DIAGIN,DIAG1,DIAG2,ONE,SDIAG1,SDIAG2
      REAL T,ZERO,ZXYMNS,ZXY1NS
C
C   CONSTRUCT CONSTANTS
C
      DATA ZERO,ONE/0.0E0,1.0E0/
C
C   DEFINE SOME VARIABLES
C
      MM1 = M - 1
      MP1 = M + 1
      NM1 = N - 1
      NP1 = N + 1
      NPM = N + M
      IERR = 2
C
C   OBTAIN Y-PARTIAL DERIVATIVES ALONG  Y = Y(1)
C
      DELY1 = DELY
      IF (IOPTY .EQ. 0) DELY1 = Y(2) - Y(1)
      IF (DELY1 .LE. ZERO) GO TO 9000
      IF (SIGMAY*DELY1 .LE. 650.0) GO TO 5
      IERR = 3
      GO TO 9000
    5 IF (ISLPSW(3) .EQ. 2) GO TO 100
      IF (ISLPSW(3) .EQ. 1) GO TO 20
C
C   THIS CODE IS FOR A USER-DEFINED EDGE CONDITION
```

```
      DO 10 I = 1,M
      ZP(I,1,1) = ZY1(I)
   10 CONTINUE
      IF (ISLPSW(4) .NE. 1) GO TO 100
      IF (IOPTY .EQ. 0) GO TO 140
C
C     THIS CODE IS FOR A COMPUTER-DEFINED EDGE CONDITION
C
   20 DELY2 = DELY1 + DELY1
      IF (IOPTY .EQ. 0 .AND. N .GT. 2) DELY2 = Y(3) - Y(1)
      IF (DELY2 .LE. DELY1) GO TO 9000
      CALL QXZ060(DELY1,DELY2,SIGMAY,C1,C2,C3,N)
C
C     BRANCH SINCE ONLY THE  CI'S  ARE NEEDED
C
      IF (ISLPSW(3) .EQ. 0) GO TO 130
      DO 30 I = 1,M
      ZP(I,1,1) = C1*Z(I,1) + C2*Z(I,2)
   30 CONTINUE
      IF (N .EQ. 2) GO TO 100
      DO 40 I = 1,M
      ZP(I,1,1) = ZP(I,1,1) + C3*Z(I,3)
   40 CONTINUE
C
C     OBTAIN Y-PARTIAL DERIVATIVES ALONG  Y  =  Y(N)
C
  100 IF (ISLPSW(4) .EQ. 2) GO TO 200
      IF (ISLPSW(4) .EQ. 1) GO TO 120
C
C     THIS CODE IS FOR A USER-DEFINED EDGE CONDITION
      DO 110 I = 1,M
      TEMP(N+I) = ZYN(I)
  110 CONTINUE
      GO TO 200
C
C     THIS CODE IS FOR A COMPUTER-DEFINED EDGE CONDITION
  120 IF (IOPTY .EQ. 0) GO TO 140
C
C     THIS CODE IS FOR THE Y-EQUALLY-SPACED CASE
C
  130 C1 = -C1
      C2 = -C2
      IF (N .GT. 2) C3 = -C3
      GO TO 150
C
C     THIS CODE IS FOR THE NON-Y-EQUALLY-SPACED CASE
C
  140 DELYN = Y(N) - Y(NM1)
      IF (DELYN .LE. ZERO) GO TO 9000
      IF (SIGMAY*DELYN .LE. 650.0) GO TO 145
      IERR = 3
      GO TO 9000
  145 DELYNM = DELYN + DELYN
      IF (N .GT. 2) DELYNM = Y(N) - Y(N-2)
      IF (DELYNM .LE. DELYN) GO TO 9000
      CALL QXZ060(-DELYN,-DELYNM,SIGMAY,C1,C2,C3,N)
  150 DO 160 I = 1,M
      TEMP(N+I) = C1*Z(I,N) + C2*Z(I,NM1)
  160 CONTINUE
      IF (N .EQ. 2) GO TO 200
      DO 170 I = 1,M
      NPI = N + I
      TEMP(NPI) = TEMP(NPI) + C3*Z(I,N-2)
  170 CONTINUE
C
C     OBTAIN X-PARTIAL DERIVATIVES ALONG X = X(1)
```

```
C
  200 DELX1 = DELX
      IF (IOPTX .EQ. 0) DELX1 = X(2) - X(1)
      IF (DELX1 .LE. ZERO) GO TO 9000
      IF (SIGMAX*DELX1 .LE. 650.0) GO TO 205
      IERR = 3
      GO TO 9000
  205 IF (ISLPSW(1) .EQ. 2) GO TO 400
      IF (ISLPSW(1) .EQ. 1) GO TO 220
C
C   THIS CODE IS FOR A USER-DEFINED EDGE CONDITION
C
      DO 210 J = 1,N
      ZP(1,J,2) = ZX1(J)
  210 CONTINUE
      IF (ISLPSW(5) .EQ. 1) GO TO 220
      IF (ISLPSW(7) .EQ. 1) GO TO 220
      IF (IOPTX .EQ. 0) GO TO 310
      IF (ISLPSW(2) .EQ. 1) GO TO 220
      IF (ISLPSW(6) .EQ. 1) GO TO 220
      IF (ISLPSW(8) .EQ. 0) GO TO 310
C
C   THIS CODE IS FOR A COMPUTER-DEFINED EDGE CONDITION
C
  220 DELX2 = DELX1 + DELX1
      IF (IOPTX .EQ. 0 .AND. M .GT. 2) DELX2 = X(3) - X(1)
      IF (DELX2 .LE. DELX1) GO TO 9000
      CALL QXZ060(DELX1,DELX2,SIGMAX,C1,C2,C3,M)
C
C   BRANCH SINCE ONLY THE  C1'S  ARE NEEDED
C
      IF (ISLPSW(1) .EQ. 0) GO TO 300
      DO 230 J = 1,N
      ZP(1,J,2) = C1*Z(1,J) + C2*Z(2,J)
  230 CONTINUE
      IF (M .EQ. 2) GO TO 300
      DO 240 J = 1,N
      ZP(1,J,2) = ZP(1,J,2) + C3*Z(3,J)
  240 CONTINUE
C
C   OBTAIN X-Y-PARTIAL DERIVATIVE AT  (X(1),Y(1))
C
  300 IF (ISLPSW(5) .EQ. 1) GO TO 320
C
C   THIS CODE IS FOR A USER-DEFINED CORNER CONDITION
C
  310 ZP(1,1,3) = ZXY11
      GO TO 330
C
C   THIS CODE IS FOR A COMPUTER-DEFINED CORNER CONDITION
C
  320 ZP(1,1,3) = C1*ZP(1,1,1) + C2*ZP(2,1,1)
      IF (M .GT. 2) ZP(1,1,3) = ZP(1,1,3) + C3*ZP(3,1,1)
C
C   OBTAIN X-Y-PARTIAL DERIVATIVE AT  (X(1),Y(N))
C
  330 IF (ISLPSW(7) .EQ. 1) GO TO 340
C
C   THIS CODE IS FOR A USER-DEFINED CORNER CONDITION
      ZXY1NS = ZXY1N
      GO TO 500
C
C   THIS CODE IS FOR A COMPUTER-DEFINED CORNER CONDITION
C
  340 ZXY1NS = C1*TEMP(N+1) + C2*TEMP(N+2)
      IF (M .GT. 2) ZXY1NS = ZXY1NS + C3*TEMP(N+3)
```

```
      GO TO 500
C
C   THIS CODE IS FOR A NATURAL EDGE ALONG  X  =  X(1)
C
  400 DO 410 J=1,N
      ZP(1,J,2) = ZERO
  410 CONTINUE
      ZP(1,1,3) = ZERO
      ZXYINS = ZERO
C
C   OBTAIN X-PARTIAL DERIVATIVE ALONG  X  =  X(M)
C
  500 IF (ISLPSW(2) .EQ. 2) GO TO 700
      IF (ISLPSW(2) .EQ. 1) GO TO 520
C
C   THIS CODE IS FOR A USER-DEFINED EDGE CONDITION
C
      DO 510 J = 1,N
      TEMP(NPM+J) = ZXM(J)
  510 CONTINUE
      IF (ISLPSW(6) .EQ. 1) GO TO 520
      IF (ISLPSW(8) .EQ. 0) GO TO 610
C
C   THIS CODE IS FOR A COMPUTER-DEFINED EDGE CONDITION
C
  520 IF (IOPTX .EQ. 0) GO TO 530
C
C   THIS CODE IS FOR THE X-EQUALLY-SPACED CASE
C
      C1 = -C1
      C2 = -C2
      IF (M .GT. 2) C3 = -C3
      GO TO 540
C
C   THIS CODE IS FOR THE NON-X-EQUALLY-SPACED CASE
C
  530 DELXM = X(M) - X(MM1)
      IF (DELXM .LE. ZERO) GO TO 9000
      IF (SIGMAX*DELXM .LE. 650.0) GO TO 535
      IERR = 3
      GO TO 9000
  535 DELXMM = DELXM + DELXM
      IF (M .GT. 2) DELXMM = X(M) - X(M-2)
      IF (DELXMM .LE. DELXM) GO TO 9000
      CALL QXZ060(-DELXM,-DELXMM,SIGMAX,C1,C2,C3,M)
C
C   BRANCH SINCE ONLY THE  CI'S  ARE NEEDED
C
  540 IF (ISLPSW(2) .EQ. 0) GO TO 600
      DO 550 J = 1,N
      TEMP(NPM+J) = C1*Z(M,J) + C2*Z(MM1,J)
  550 CONTINUE
      IF (M .EQ. 2) GO TO 600
      DO 560 J = 1,N
      NPMPJ = NPM + J
      TEMP(NPMPJ) = TEMP(NPMPJ) + C3*Z(M-2,J)
  560 CONTINUE
C
C   OBTAIN X-Y-PARTIAL DERIVATIVE AT  (X(M),Y(1))
C
  600 IF (ISLPSW(6) .EQ. 1) GO TO 620
C
C   THIS CODE IS FOR A USER-DEFINED CORNER CONDITION
C
  610 ZP(M,1,3) = ZXYM1
      GO TO 630
```

```
C
C   THIS CODE IS FOR A COMPUTER-DEFINED CORNER CONDITION
C
  620 ZP(M,1,3) = C1*ZP(M,1,1) + C2*ZP(MM1,1,1)
      IF (M .GT. 2) ZP(M,1,3) = ZP(M,1,3) + C3*ZP(M-2,1,1)
C
C   OBTAIN X-Y-PARTIAL DERIVATIVE AT  (X(M),Y(N))
C
  630 IF (ISLPSW(8) .EQ. 1) GO TO 640
C
C   THIS CODE IS FOR A USER-DEFINED CORNER CONDITION
C
      ZXYMNS = ZXYMN
      GO TO 800
C
C   THIS CODE IS FOR A COMPUTER-DEFINED CORNER CONDITION
C
  640 ZXYMNS = C1*TEMP(NPM) + C2*TEMP(NPM-1)
      IF (M .GT. 2) ZXYMNS = ZXYMNS + C3*TEMP(NPM-2)
      GO TO 800
C
C   THIS CODE IS FOR A NATURAL EDGE ALONG  X  =  X(M)
C
  700 DO 710 J=1,N
      TEMP(NPM+J) = ZERO
  710 CONTINUE
      ZP(M,1,3) = ZERO
      ZXYMNS = ZERO
C
C   SET UP RIGHT HAND SIDES AND TRIDIAGONAL SYSTEM FOR Y-GRID
C   PERFORM FORWARD ELIMINATION ON THE FIRST COLUMN
C
  800 DEL1 = DELY1
      DELI = ONE/DEL1
      DO 810 I = 1,M
      ZP(I,2,1) = DELI*(Z(I,2) - Z(I,1))
  810 CONTINUE
      ZP(1,2,3) = DELI*(ZP(1,2,2) - ZP(1,1,2))
      ZP(M,2,3) = DELI*(TEMP(NPM+2) - TEMP(NPM+1))
      CALL QXZ061 (DIAG1,SDIAG1,SIGMAY,DEL1)
      DIAGI = ONE/DIAG1
      IF (ISLPSW(3) .EQ. 2) GO TO 830
C
C   THIS CODE IS FOR A NON-NATURAL EDGE ALONG  Y  =  Y(1)
C
      ZP(1,1,3) = DIAGI*(ZP(1,2,3) - ZP(1,1,3))
      ZP(M,1,3) = DIAGI*(ZP(M,2,3) - ZP(M,1,3))
      TEMP(1) = DIAGI*SDIAG1
      DO 820 I = 1,M
      ZP(I,1,1) = DIAGI*(ZP(I,2,1) - ZP(I,1,1))
  820 CONTINUE
      GO TO 850
C
C   THIS CODE IS FOR A NATURAL EDGE ALONG  Y  =  Y(1)
C
  830 DO 840 I=1,M
      ZP(I,1,1) = ZERO
  840 CONTINUE
      ZP(1,1,3) = ZERO
      ZP(M,1,3) = ZERO
      TEMP(1) = ZERO
  850 IF (N .EQ. 2) GO TO 1000
C
C   THESE TERMS ARE DEFINED IN THE Y-EQUALLY-SPACED CASE
C
      DEL2 = DEL1
```

```
      DIAG2 = DIAG1
      SDIAG2 = SDIAG1
C
C   THIS IS THE FORWARD ELIMINATION LOOP
C   LET THE WILD RUMPUS BEGIN
C
      DO 930 J = 2,NM1
      JM1 = J - 1
      JP1 = J + 1
      NPMPJ = NPM + J
      IF (IOPTY .NE. 0) GO TO 900
C
C   THIS CODE IS FOR THE NON-Y-EQUALLY-SPACED CASE
C
      DEL2 = Y(JP1) - Y(J)
      IF (DEL2 .LE. ZERO) GO TO 9000
      IF (SIGMAY*DEL2 .LE. 650.0) GO TO 890
      IERR = 3
      GO TO 9000
  890 DELI = ONE/DEL2
      CALL QXZ061(DIAG2,SDIAG2,SIGMAY,DEL2)
C
C   CONTINUE THE FORWARD ELIMINATION LOOP
C
  900 DO 910 I = 1,M
      ZP(I,JP1,1) = DELI*(Z(I,JP1) - Z(I,J))
  910 CONTINUE
      ZP(1,JP1,3) = DELI*(ZP(1,JP1,2) - ZP(1,J,2))
      ZP(M,JP1,3) = DELI*(TEMP(NPMPJ+1) - TEMP(NPMPJ))
      DIAGIN = ONE/(DIAG1 + DIAG2 - SDIAG1*TEMP(JM1))
      DO 920 I = 1,M
      ZP(I,J,1) = DIAGIN*(ZP(I,JP1,1) - ZP(I,J,1) -
     *            SDIAG1*ZP(I,JM1,1))
  920 CONTINUE
      ZP(1,J,3) = DIAGIN*(ZP(1,JP1,3) - ZP(1,J,3) -
     *            SDIAG1*ZP(1,JM1,3))
      ZP(M,J,3) = DIAGIN*(ZP(M,JP1,3) - ZP(M,J,3) -
     *            SDIAG1*ZP(M,JM1,3))
      TEMP(J) = DIAGIN*SDIAG2
      DIAG1 = DIAG2
      SDIAG1 = SDIAG2
  930 CONTINUE
C
C   END OF THE FORWARD ELIMINATION LOOP
C   NOW PERFORM FORWARD ELIMINATION ON THE LAST COLUMN
C
 1000 IF (ISLPSW(4) .EQ. 2) GO TO 1020
C
C   THIS CODE IS FOR A NON-NATURAL EDGE ALONG  Y  =  Y(N)
C
      DIAGIN = ONE/(DIAG1 - SDIAG1*TEMP(NM1))
      ZP(1,N,3) = DIAGIN*(ZXY1NS - ZP(1,N,3) -
     *            SDIAG1*ZP(1,NM1,3))
      TEMP(N) = DIAGIN*(ZXYMNS - ZP(M,N,3) -
     *          SDIAG1*ZP(M,NM1,3))
      DO 1010 I = 1,M
      ZP(I,N,1) = DIAGIN*(TEMP(N+I) - ZP(I,N,1) -
     *            SDIAG1*ZP(I,NM1,1))
 1010 CONTINUE
      GO TO 1100
C
C   THIS CODE IS FOR A NATURAL EDGE ALONG  Y  =  Y(N)
C
 1020 DO 1030 I=1,M
      ZP(I,N,1) = ZERO
 1030 CONTINUE
```

```
      ZP(1,N,3) = ZERO
      TEMP(N) = ZERO
C
C   PERFORM BACK SUBSTITUTION
C
 1100 DO 1120 J = 2,N
      JBAK = NP1 - J
      JBAKP1 = JBAK + 1
      T = TEMP(JBAK)
      DO 1110 I = 1,M
      ZP(I,JBAK,1) = ZP(I,JBAK,1) - T*ZP(I,JBAKP1,1)
 1110 CONTINUE
      ZP(1,JBAK,3) = ZP(1,JBAK,3) - T*ZP(1,JBAKP1,3)
      TEMP(JBAK) = ZP(M,JBAK,3) - T*TEMP(JBAKP1)
 1120 CONTINUE
C
C   SET UP RIGHT HAND SIDES AND TRIDIAGONAL SYSTEM FOR X-GRID
C   PERFORM FORWARD ELIMINATION ON THE FIRST COLUMN
C   RECALL THAT SOME TERMS ARE ALREADY ZERO FOR A NATURAL EDGE
C
      DEL1 = DELX1
      DELI = ONE/DEL1
      DO 1200 J = 1,N
      ZP(2,J,2) = DELI*(Z(2,J) - Z(1,J))
      ZP(2,J,3) = DELI*(ZP(2,J,1) - ZP(1,J,1))
 1200 CONTINUE
      CALL QXZ061 (DIAG1,SDIAG1,SIGMAX,DEL1)
      DIAGI = ONE/DIAG1
      IF (ISLPSW(1) .EQ. 2) GO TO 1220
      DO 1210 J = 1,N
      ZP(1,J,2) = DIAGI*(ZP(2,J,2) - ZP(1,J,2))
      ZP(1,J,3) = DIAGI*(ZP(2,J,3) - ZP(1,J,3))
 1210 CONTINUE
      TEMP(NP1) = DIAGI*SDIAG1
 1220 IF (ISLPSW(1) .EQ. 2) TEMP(NP1) = ZERO
      IF (M .EQ. 2) GO TO 1400
C
C   THESE TERMS ARE DEFINED IN THE X-EQUALLY-SPACED CASE
C
      DEL2 = DEL1
      DIAG2 = DIAG1
      SDIAG2 = SDIAG1
C
C   THIS IS THE FORWARD ELIMINATION LOOP
C
      DO 1330 I = 2,MM1
      IM1 = I - 1
      IP1 = I + 1
      NPI = N + I
      IF (IOPTX .NE. 0) GO TO 1300
C
C   THIS CODE IS FOR THE NON-X-EQUALLY-SPACED CASE
C
      DEL2 = X(IP1) - X(I)
      IF (DEL2 .LE. ZERO) GO TO 9000
      IF (SIGMAX*DEL2 .LE. 650.0) GO TO 1290
      IERR = 3
      GO TO 9000
 1290 DELI = ONE/DEL2
      CALL QXZ061(DIAG2,SDIAG2,SIGMAX,DEL2)
C
C   CONTINUE THE FORWARD ELIMINATION LOOP
C
 1300 DO 1310 J = 1,N
      ZP(IP1,J,2) = DELI*(Z(IP1,J) - Z(I,J))
      ZP(IP1,J,3) = DELI*(ZP(IP1,J,1) - ZP(I,J,1))
```

```
 1310 CONTINUE
      DIAGIN = ONE/(DIAG1 + DIAG2  -  SDIAG1*TEMP(NPI - 1))
      DO 1320 J = 1,N
      ZP(I,J,2) = DIAGIN*(ZP(IP1,J,2) - ZP(I,J,2) -
     *            SDIAG1*ZP(IM1,J,2))
      ZP(I,J,3) = DIAGIN*(ZP(IP1,J,3) - ZP(I,J,3) -
     *            SDIAG1*ZP(IM1,J,3))
 1320 CONTINUE
      TEMP(NPI) = DIAGIN*SDIAG2
      DIAG1 = DIAG2
      SDIAG1 = SDIAG2
 1330 CONTINUE
C
C   END OF THE FORWARD ELIMINATION LOOP
C   NOW PERFORM FORWARD ELIMINATION ON THE LAST COLUMN
C
 1400 IF (ISLPSW(2) .EQ. 2) GO TO 1420
C
C   THIS CODE IS FOR A NON-NATURAL EDGE ALONG  X  =  X(M)
C
      DIAGIN = ONE/(DIAG1 - SDIAG1*TEMP(NPM - 1))
      DO 1410 J = 1,N
      ZP(M,J,2) = DIAGIN*(TEMP(NPM+J) - ZP(M,J,2) -
     *            SDIAG1*ZP(MM1,J,2))
      ZP(M,J,3) = DIAGIN*(TEMP(J) - ZP(M,J,3) -
     *            SDIAG1*ZP(MM1,J,3))
 1410 CONTINUE
      GO TO 1500
C
C   THIS CODE IS FOR A NATURAL EDGE ALONG  X  =  X(M)
C
 1420 DO 1430 J=1,N
      ZP(M,J,2) = ZERO
      ZP(M,J,3) = ZERO
 1430 CONTINUE
C
C   PERFORM BACK SUBSTITUTION
C
 1500 DO 1520 I = 2,M
      IBAK = MP1 - I
      IBAKP1 = IBAK + 1
      T = TEMP(N+IBAK)
      DO 1510 J = 1,N
      ZP(IBAK,J,2) = ZP(IBAK,J,2) - T*ZP(IBAKP1,J,2)
      ZP(IBAK,J,3) = ZP(IBAK,J,3) - T*ZP(IBAKP1,J,3)
 1510 CONTINUE
 1520 CONTINUE
      IERR = 0
 9000 RETURN
      END
      GO TO 1500
C
C   THIS CODE IS FOR A NATURAL EDGE ALONG  X  =  X(M)
C
 1420 DO 1430 J=1,N
      ZP(M,J,2) = ZERO
      ZP(M,J,3) = ZERO
 1430 CONTINUE
C
C   PERFORM BACK SUBSTITUTION
C
 1500 DO 1520 I = 2,M
      IBAK = MP1 - I
      IBAKP1 = IBAK + 1
      T = TEMP(N+IBAK)
      DO 1510 J = 1,N
```